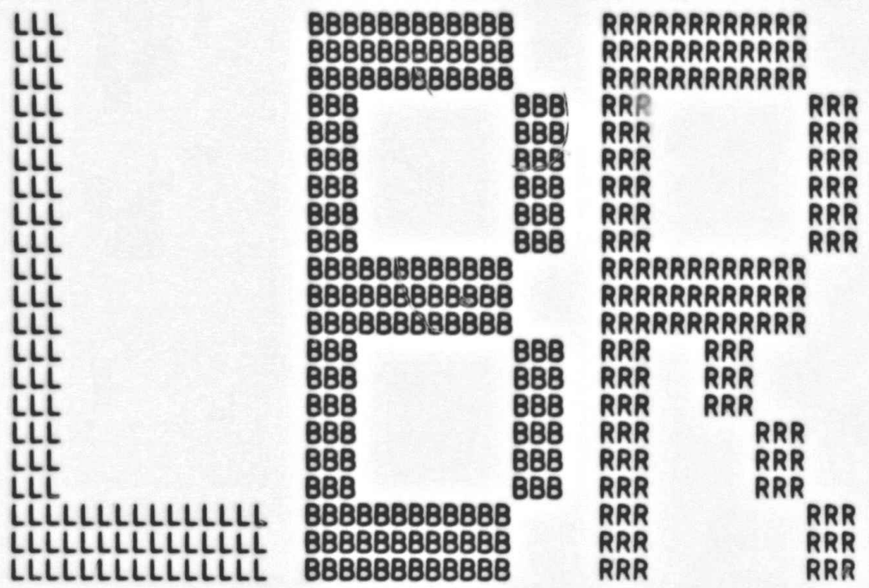

_$2

Val

002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
00F
00F
00F
00F
00F
00F
00F
00F
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF

FILEID**OLDLIB

```
  OOOOOO    LL          DDDDDDDD    LL          IIIIIII   BBBBBBBB
  OOOOOO    LL          DDDDDDDD    LL          IIIIIII   BBBBBBBB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BBBBBBBB
OO      OO  LL          DD      DD  LL            II      BBBBBBBB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BB      BB
OO      OO  LL          DD      DD  LL            II      BB      BB
  OOOOOO    LLLLLLLLLL  DDDDDDDD    LLLLLLLLLL  IIIIIII   BBBBBBBB
  OOOOOO    LLLLLLLLLL  DDDDDDDD    LLLLLLLLLL  IIIIIII   BBBBBBBB
```

```
LL            IIIIIII     SSSSSSSS
LL            IIIIIII     SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIIII   SSSSSSSS
LLLLLLLLLL    IIIIIII   SSSSSSSS
```

LBR
V04

```
;     1        0001  0 MODULE LBR_OLDLIB (                        ! Routines to process old format libraries
;     2        0002  0                 LANGUAGE (BLISS32),
;     3        0003  0                 IDENT = 'V04-000'
;     4        0004  0                 ) =
;     5        0005  1 BEGIN
;     6        0006  1
;     7        0007  1 !
;     8        0008  1 !*****************************************************************************
;     9        0009  1 !*                                                                           *
;    10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12        0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13        0013  1 !*                                                                           *
;    14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
;    15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;    16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19        0019  1 !*  TRANSFERRED.                                                              *
;    20        0020  1 !*                                                                           *
;    21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23        0023  1 !*  CORPORATION.                                                              *
;    24        0024  1 !*                                                                           *
;    25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27        0027  1 !*                                                                           *
;    28        0028  1 !*                                                                           *
;    29        0029  1 !*****************************************************************************
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 !
;    33        0033  1 ! FACILITY:  Library access procedures
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !
;    37        0037  1 !               The VAX/VMS librarian procedures implement a standard access method
;    38        0038  1 !               to libraries through a shared, common procedure set.
;    39        0039  1 !
;    40        0040  1 ! ENVIRONMENT:
;    41        0041  1 !
;    42        0042  1 !       VAX native, user mode.
;    43        0043  1 !
;    44        0044  1 !--
;    45        0045  1 !
;    46        0046  1 !
;    47        0047  1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  24-July-1979
;    48        0048  1 !
;    49        0049  1 ! MODIFIED BY:
;    50        0050  1 !
;    51        0051  1 !       V02-001         RPG0001         Bob Grosso      31-Aug-1981
;    52        0052  1 !               Remove LBRMSG.
;    53        0053  1 !--
;    54        0054  1
;    55        0055  1
```

```
;    57          0056  1 LIBRARY
;    58          0057  1         'SYS$LIBRARY:STARLET.L32';
;    59          0058  1 REQUIRE
;    60          0059  1         'PREFIX';
;    61          0198  1 REQUIRE
;    62          0199  1         'LBRDEF';
;    63          0790  1 REQUIRE
;    64          0791  1         'OLDFMTDEF';
;    65          0887  1
;    66          0888  1 LINKAGE
;    67          0889  1     fmg_match = JSB (REGISTER = 2, REGISTER = 3, REGISTER = 4, REGISTER = 5) : NOTUSED (10, 11);
;    68          0890  1
;    69          0891  1 FORWARD ROUTINE
;    70          0892  1     check_wild,                                 !Check wildcard match
;    71          0893  1     call_user,                                  !Call user routine
;    72          0894  1     check_rfa,                                  !Check RFA before calling user
;    73          0895  1     travers_old_idx;                            !Traverse index
;    74          0896  1
;    75          0897  1 EXTERNAL ROUTINE
;    76          0898  1     fmg$match_name : fmg_match,                 !Embedded wild card matching
;    77          0899  1     find_block : JSB_3,                         !Find block and map in memory
;    78          0900  1     read_block : JSB_2,                         !Read disk block
;    79          0901  1     read_n_block : JSB_2,                       !Read and cache multiple disk blocks
;    80          0902  1     get_mem : JSB_2,                            !Allocate dynamic memory
;    81          0903  1     dealloc_mem : JSB_2;                        !Deallocate dynamic memory
;    82          0904  1
;    83          0905  1 EXTERNAL
;    84          0906  1     lbr$gl_control : REF BBLOCK;                !Librarian control block
;    85          0907  1
;    86          0908  1 EXTERNAL LITERAL
;    87          0909  1     lbr$_keynotfnd,
;    88          0910  1     lbr$_normal;
;    89          0911  1
;    90          0912  1 GLOBAL LITERAL
;    91          0913  1     lbr$k_libblocks = 10;                       !Size of window into index
```

```
;    93          0914  1 !
;    94          0915  1 ! MACROS:
;    95          0916  1 !       The following three macros are used to set the elements of a
;    96          0917  1 !       library table name descriptor relative to some other name descriptor
;    97          0918  1 !
;    98          0919  1 !       CALLING SEQUENCE (same for each):
;    99          0920  1 !               OTHER   is name of the source of the descriptor variables
;   100          0921  1 !               BLOCKOFF        is the offset to add to the relative block number field
;   101          0922  1 !               ENTRYOFF        is the offset to add to the relative entry number field
;   102          0923  1 !
;   103          0924  1 !       The code generated is conditional upon the existence of these parameters
;   104          0925  1 !
;   105          0926  1 MACRO
;   106        M 0927  1         SETHILIMIT(OTHER,BLOCKOFF,ENTRYOFF) =           ! Set high limit name descriptor
;   107          0928  1                 SETENTRYDESC(HILIMIT,OTHER,BLOCKOFF,ENTRYOFF)%,
;   108          0929  1 !
;   109        M 0930  1         SETLOLIMIT(OTHER,BLOCKOFF,ENTRYOFF) =           ! Set low limit name descriptor
;   110          0931  1                 SETENTRYDESC(LOLIMIT,OTHER,BLOCKOFF,ENTRYOFF)%,
;   111          0932  1 !
;   112        M 0933  1         SETRIAL(OTHER,BLOCKOFF,ENTRYOFF) =              ! Set the trial name descriptor
;   113          0934  1                 SETENTRYDESC(TRIAL,OTHER,BLOCKOFF,ENTRYOFF)%,
;   114          0935  1 !
;   115          0936  1 !       This macro does all the work for the above three macros.
;   116          0937  1 !       The calling arguments are the same except for the name of the descriptor
;   117          0938  1 !       whose elements are to be set.
;   118          0939  1 !       The following compile time actions are taken (in order):
;   119          0940  1 !               if "BLOCKOFF" is specified:
;   120          0941  1 !                       "ENTRYOFF" is taken as a general expression
;   121          0942  1 !                       as is "BLOCKOFF" and the entry and rel. block fields of
;   122          0943  1 !                       the desired descriptor are set.
;   123          0944  1 !               If both are null, the entry, relative block and table address are
;   124          0945  1 !                       just copied.
;   125          0946  1 !
;   126          0947  1 !               If only "BLOCKOFF" is null, "ENTRYOFF" is assumed to be
;   127          0948  1 !                       either + or -1. Code is generated to increment or decrement
;   128          0949  1 !                       the entry number with appropriate checks
;   129          0950  1 !                       for movement off top or bottom of the block
;   130          0951  1 !
;   131        M 0952  1         SETENTRYDESC(THISONE,OTHER,BLOCKOFF,ENTRYOFF) =
;   132        M 0953  1             %IF NOT %NULL(BLOCKOFF)                     ! Fully general expressions assumed so
;   133        M 0954  1             %THEN       %NAME(THISONE,'RBN') = BLOCKOFF;        ! Set the block offset
;   134        M 0955  1                       %NAME(THISONE,'ENT') = ENTRYOFF;        ! And the entry number
;   135        M 0956  1                                                       !
;   136        M 0957  1             %ELSE %IF %NULL(ENTRYOFF)                         ! If both arguments are null so
;   137        M 0958  1                   %THEN CH$MOVE(12, %NAME(OTHER,'ENT'), %NAME(THISONE,'ENT')); !  copy entry number, rel blk
;   138        M 0959  1                                                               !
;   139        M 0960  1                   %ELSE %IF NOT %IDENTICAL(THISONE,OTHER)             ! If not the same
;   140        M 0961  1                         %THEN %NAME(THISONE,'RBN') = .%NAME(OTHER,'RBN');     ! Copy block number
;   141        M 0962  1                         %FI
;   142        M 0963  1                         %IF %IDENTICAL(ENTRYOFF,-1)                   ! If the entry number
;   143        M 0964  1                         %THEN IF (%NAME(THISONE,'ENT') = .%NAME(OTHER,'ENT')  ! Is being decremented
;   144        M 0965  1                                   + (ENTRYOFF)) LSS 0
;   145        M 0966  1                               THEN BEGIN                          ! Do it checking for crossin
;   146        M 0967  1                                       %NAME(THISONE,'ENT') = ENTSPERBLK - 1;  ! Block down and if so
;   147        M 0968  1                                       %NAME(THISONE,'RBN') = .%NAME(OTHER,    ! Set to top of previous
;   148        M 0969  1                                                               'RBN') - 1;     !
;   149        M 0970  1                                   END;                                ! Otherwise leave rbn as is
```

```
;  150        M 0971  1
;  151        M 0972  1                    %ELSE %IF %IDENTICAL(ENTRYOFF,1)                                    !
;  152        M 0973  1                          %THEN IF (%NAME(THISONE,'ENT') = .%NAME(OTHER,                ! If incrementing the
;  153        M 0974  1                                                  'ENT') + (ENTRYOFF)) GEQ              !
;  154        M 0975  1                                                      ENTSPERBLK                        !
;  155        M 0976  1                               THEN BEGIN                                               ! Entry then check for block
;  156        M 0977  1                                    %NAME(THISONE,'ENT') = 0;                           ! Crossing up to next
;  157        M 0978  1                                    %NAME(THISONE,'RBN') =                              ! And set to its first
;  158        M 0979  1                                                .%NAME(OTHER,'RBN') + 1;                !
;  159        M 0980  1                               END;
;  160        M 0981  1
;  161        M 0982  1                          %ELSE IF (%NAME(THISONE,'ENT') = .%NAME(OTHER,'ENT')          ! If a general expre
;  162        M 0983  1                               + (ENTRYOFF)) LSS 0                                      ! Then have to check both
;  163        M 0984  1                               THEN BEGIN                                               ! Do it checking for crossin
;  164        M 0985  1                                    %NAME(THISONE,'ENT') = ENTSPERBLK - 1;              ! Block down and if so
;  165        M 0986  1                                    %NAME(THISONE,'RBN') = .%NAME(OTHER,                ! Set to top of previous
;  166        M 0987  1                                                           'RBN') - 1;                  !
;  167        M 0988  1                                END                                                     ! Otherwise leave rbn as is
;  168        M 0989  1                               ELSE IF .%NAME(THISONE,'ENT') GEQ ENTSPERBLK
;  169        M 0990  1                                    THEN BEGIN
;  170        M 0991  1                                         %NAME(THISONE,'ENT') = 0;                      ! Crossing up to next
;  171        M 0992  1                                         %NAME(THISONE,'RBN') =                         ! And set to its first
;  172        M 0993  1                                                     .%NAME(OTHER,'RBN') + 1;           !
;  173        M 0994  1                                    END;
;  174        M 0995  1                          %FI
;  175        M 0996  1                     %FI
;  176        M 0997  1                %FI
;  177        M 0998  1           %FI
;  178        M 0999  1           %IF NOT %NULL(BLOCKOFF)
;  179        M 1000  1            OR NOT %NULL(OTHER)
;  180        M 1001  1           %THEN
;  181        M 1002  1               BEGIN
;  182        M 1003  1                   perform(find_block(.windowbaseblk+.%NAME(THISONE,'RBN'),
;  183        M 1004  1                                      blockaddr, cache_entry));
;  184        M 1005  1                   %NAME(THISONE,'ADR') = .blockaddr + entrysize*.%NAME(THISONE,'ENT');
;  185        M 1006  1                   END;
;  186        M 1007  1           %FI
;  187           1008  1      %;
```

```
;  189          1009  1 GLOBAL ROUTINE lbr_old_lib_dat (header) =
;  190          1010  2 BEGIN
;  191          1011  2 !
;  192          1012  2 ! This routine extracts the needed information from the library
;  193          1013  2 ! header of an old format (VMS R1) library and stores it in a
;  194          1014  2 ! block of memory (the last part of the library header)
;  195          1015  2 !
;  196          1016  2
;  197          1017  2 MAP
;  198          1018  2     header : REF BBLOCK;
;  199          1019  2
;  200          1020  2 BIND
;  201          1021  2     oldctx = header[ohd$t_oldctx] : BBLOCK;
;  202          1022  2
;  203          1023  2 LOCAL
;  204          1024  2     index_desc : REF BBLOCK;
;  205          1025  2
;  206          1026  2 CH$FILL(0, ofl$c_length, oldctx);                    !Zero the block
;  207          1027  2 oldctx[ofl$l_mntvbn] = .header[ohd$w_mntvbn];         !VBN of start of MNT
;  208          1028  2 oldctx[ofl$l_mntesiz] = .header[ohd$w_mntesiz];       !Size of MNT entry
;  209          1029  2 oldctx[ofl$l_numods] = .header[ohd$w_mntallo] -       !compute number of modules
;  210          1030  2                       .header[ohd$w_mntaval];
;  211          1031  2 IF .oldctx[ofl$l_mntesiz] NEQ 0
;  212          1032  3 THEN BEGIN
;  213          1033  3     oldctx[ofl$l_mntepblk] =
;  214          1034  3                 [br$c_pagesize/.oldctx[ofl$l_mntesiz]; !Number entries/block
;  215          1035  4     oldctx[ofl$l_mntblks] = (.oldctx[ofl$l_numods]
;  216          1036  4                             + .oldctx[ofl$l_mntepblk] - 1)
;  217          1037  3                             / .oldctx[ofl$l_mntepblk];
;  218          1038  2     END;
;  219          1039  2 oldctx[ofl$l_gstvbn] = .header[ohd$w_gstvbn];         !VBN of start of GST
;  220          1040  2 oldctx[ofl$l_gstesiz] = .header[ohd$w_gstesiz];       !Size of GST entry
;  221          1041  2 oldctx[ofl$l_numsyms] = .header[ohd$w_gstallo] -
;  222          1042  2                       .header[ohd$w_gstaval];
;  223          1043  2 IF .oldctx[ofl$l_gstesiz] NEQ 0
;  224          1044  3 THEN BEGIN
;  225          1045  3     oldctx[ofl$l_gstepblk] =
;  226          1046  3                 [br$c_pagesize/.oldctx[ofl$l_gstesiz];  !Entries/block
;  227          1047  4     oldctx[ofl$l_gstblks] = (.oldctx[ofl$l_numsyms]
;  228          1048  4                             + .oldctx[ofl$l_gstepblk] - 1)
;  229          1049  3                             / .oldctx[ofl$l_gstepblk];
;  230          1050  2     END;
;  231          1051  2 !
;  232          1052  2 ! Set the number of indices into the header location lhd$b_nindex.
;  233          1053  2 ! Note that this will overwrite the byte ohd$b_fmtlvl, which is a
;  234          1054  2 ! constant used for sanity checkin and is not needed after this point.
;  235          1055  2 !
;  236          1056  3 header[lhd$b_nindex] = (IF .header[ohd$b_type] EQL lbr$c_typ_obj
;  237          1057  3                             THEN 2
;  238          1058  3                             ELSE 1
;  239          1059  2                         );
;  240          1060  2 !
;  241          1061  2 ! Set the size of the module header user data into the header location
;  242          1062  2 ! lhd$b_mhdusz.  This lies in the reserved space in the memory-resident
;  243          1063  2 ! header.
;  244          1064  2 !
;  245          1065  3 header[lhd$b_mhdusz] = (IF .header[ohd$b_type] EQL lbr$c_typ_obj
```

```
;  246          1066  3                            THEN ofl$c_maxsymlng + 2
;  247          1067  3                            ELSE 0
;  248          1068  2                            );
;  249          1069  2  !
;  250          1070  2  ! Set the total number of index entries into the header location
;  251          1071  2  ! lhd$l_idxcnt.  This lies in the reserved space in the memory-resident
;  252          1072  2  ! header.
;  253          1073  2  !
;  254          1074  2  header [lhd$l_idxcnt] = .oldctx [ofl$l_numods] + .oldctx [ofl$l_numsyms];
;  255          1075  2  header [lhd$l_modcnt] = .oldctx [ofl$l_numods];
;  256          1076  2  !
;  257          1077  2  ! Set up phony index descriptors with only the keylen field filled in.
;  258          1078  2  ! This is so make_upper_case will work.
;  259          1079  2  !
;  260          1080  2  index_desc = .header + lhd$c_idxdesc;   !Point to first descriptor
;  261          1081  2  index_desc [idd$w_keylen] = ofl$c_maxsymlng;
;  262          1082  2  index_desc [idd$w_flags] = idd$m_ascii; !Set type to ASCII
;  263          1083  2  index_desc = .index_desc + idd$c_length; !Now the second
;  264          1084  2  index_desc [idd$w_keylen] = ofl$c_maxsymlng;
;  265          1085  2  index_desc [idd$w_flags] = idd$m_ascii; !Set type to ASCII
;  266          1086  2
;  267          1087  2  RETURN true
;  268          1088  1  END;                                                !Of lbr_old_lib_dat


                                                                 .TITLE  LBR_OLDLIB
                                                                 .IDENT  \V04-000\

                                                   LBR$K_LIBBLOCKS==    10
                                                                 .EXTRN  FMG$MATCH_NAME, FIND_BLOCK
                                                                 .EXTRN  READ_BLOCK, READ_N_BLOCK
                                                                 .EXTRN  GET_MEM, DEALLOC_MEM
                                                                 .EXTRN  LBR$GL_CONTROL, LBR$_KEYNOTFND
                                                                 .EXTRN  LBR$_NORMAL

                                                                 .PSECT  $CODE$,NOWRT,2

                                          00FC 00000             .ENTRY  LBR_OLD_LIB_DAT, Save R2,R3,R4,R5,R6,R7      ; 1009
                      57        04   AC   D0 00002               MOVL    HEADER, R7                                   ; 1021
                      56      015A   C7   9E 00006               MOVAB   346(R7), R6                                  ;
  0068   8F       00  6E              00  2C 0000B               MOVC5   #0, (SP), #0, #104, (R6)                     ; 1026
                                      66     00012
                      66        1C   A7   3C 00013               MOVZWL  28(R7), (R6)                                 ; 1027
               04     A6        1A   A7   3C 00017               MOVZWL  26(R7), 4(R6)                                ; 1028
                      50        1E   A7   3C 0001C               MOVZWL  30(R7), R0                                   ; 1030
                      51        20   A7   3C 00020               MOVZWL  32(R7), R1                                   ;
         08   A6      50             51   C3 00024               SUBL3   R1, R0, 8(R6)                                ;
                                04   A6   D5 00029               TSTL    4(R6)                                        ; 1031
                                     18   13 0002C               BEQL    1$                                           ;
         10   A6 00000200  8F   04   A6   C7 0002E               DIVL3   4(R6), #512, 16(R6)                          ; 1034
              50      08   A6   10   A6   C1 00038               ADDL3   16(R6), 8(R6), R0                            ; 1036
                                     50   D7 0003E               DECL    R0                                           ; 1035
         0C   A6      50        10   A6   C7 00040               DIVL3   16(R6), R0, 12(R6)                           ; 1037
               1C     A6        14   A7   3C 00046 1$:           MOVZWL  20(R7), 28(R6)                               ; 1039
               20     A6        12   A7   3C 0004B               MOVZWL  18(R7), 32(R6)                               ; 1040
                      50        16   A7   3C 00050               MOVZWL  22(R7), R0                                   ; 1042
                      51        18   A7   3C 00054               MOVZWL  24(R7), R1                                   ;
```

```
                24   A6              50              51  C3 00058          SUBL3   R1, R0, 36(R6)                                  ;
                                         20   A6      D5 0005D          TSTL    32(R6)                                          ; 1043
                                         18   13      00060          BEQL    2$                                              ;
                2C   A6 00000200   8F    20   A6      C7 00062          DIVL3   32(R6), #512, 44(R6)                            ; 1046
                     50        24   A6    2C   A6      C1 0006C          ADDL3   44(R6), 36(R6), R0                              ; 1048
                                         50   D7      00072          DECL    R0                                              ; 1047
                28   A6              50    2C   A6      C7 00074          DIVL3   44(R6), R0, 40(R6)                              ; 1049
                                     01        67      91 0007A 2$:      CMPB    (R7), #1                                        ; 1056
                                               05      12 0007D          BNEQ    3$                                              ;
                                     50        02      D0 0007F          MOVL    #2, R0                                          ;
                                               03      11 00082          BRB     4$                                              ;
                                     50        01      D0 00084 3$:      MOVL    #1, R0                                          ;
                              01     A7        50      90 00087 4$:      MOVB    R0, 1(R7)                                       ;
                                     01        67      91 0008B          CMPB    (R7), #1                                        ; 1065
                                               05      12 0008E          BNEQ    5$                                              ;
                                     50        11      D0 00090          MOVL    #17, R0                                         ; 1066
                                               02      11 00093          BRB     6$                                              ;
                                               50      D4 00095 5$:      CLRL    R0                                              ; 1065
                              3C     A7        50      90 00097 6$:      MOVB    R0, 60(R7)                                      ;
                6A   A7       08     A6    24   A6      C1 0009B          ADDL3   36(R6), 8(R6), 106(R7)                          ; 1074
                              6E     A7    08   A6      D0 000A2          MOVL    8(R6), 110(R7)                                  ; 1075
                                     50  00C4   C7      9E 000A7          MOVAB   196(R7), INDEX_DESC                             ; 1080
                                     80 000F0001 8F     D0 000AC          MOVL    #983041, (INDEX_DESC)+                          ; 1082
                                     50        04      C0 000B3          ADDL2   #4, INDEX_DESC                                  ; 1083
                                     60 000F0001 8F     D0 000B6          MOVL    #983041, (INDEX_DESC)                           ; 1085
                                     50        01      D0 000BD          MOVL    #1, R0                                          ; 1087
                                               04      00C0              RET                                                     ; 1088

; Routine Size:  193 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  270          1089  1 GLOBAL ROUTINE lbr_old_lkp_key (keydesc, retrfa) =
;  271          1090  2 BEGIN
;  272          1091  2 !++
;  273          1092  2 ! FUNCTIONAL DESCRIPTION:
;  274          1093  2 !
;  275          1094  2 !       This routine searches one of the indices of an old format library.
;  276          1095  2 !
;  277          1096  2 ! INPUTS:
;  278          1097  2 !
;  279          1098  2 !       keyname - address of a descriptor for the key to find
;  280          1099  2 !       retrfa - address of storage to return RFA of module if found
;  281          1100  2 !
;  282          1101  2 ! ROUTINE OUTPUTS:
;  283          1102  2 !
;  284          1103  2 !       The name is found in the search table:
;  285          1104  2 !               routine value = true
;  286          1105  2 !       If name is not found routine value = false.
;  287          1106  2 !
;  288          1107  2 !
;  289          1108  2 !--
;  290          1109  2
;  291          1110  2 MAP
;  292          1111  2     keydesc : REF BBLOCK[dsc$c_s_bln],
;  293          1112  2     retrfa : REF BBLOCK[rfa$c_length];
;  294          1113  2
;  295          1114  2 BIND
;  296          1115  2     namblk = .lbr$gl_control[lbr$l_usrnam] : BBLOCK,    ! NAM block for library
;  297          1116  2     context = .lbr$gl_control[lbr$l_ctxptr] : BBLOCK,   ! Librarian context block
;  298          1117  2     header = .lbr$gl_control[lbr$l_hdrptr] : BBLOCK,    ! Library header
;  299          1118  2     oldctx = header[ohd$t_oldctx] : BBLOCK,     ! Context for old library
;  300          1119  2     idxnum = .lbr$gl_control[lbr$l_curidx] - 1, ! Index of selected index
;  301          1120  3     idxdat = (                                  ! Name the index data
;  302          1121  3                 IF idxnum EQL 0                 !
;  303          1122  3                     THEN oldctx[ofl$l_mntvbn]   ! which is the mnt if index 0
;  304          1123  3                     ELSE oldctx[ofl$l_gstvbn]   ! or the gst if index 1
;  305          1124  2             ) : BBLOCK,                         !  and is a structure
;  306          1125  2     entrysize = .idxdat[oib$l_esiz],            ! size of an entry
;  307          1126  2     entsperblk = .idxdat[oib$l_entpblk],        ! number entries in a block
;  308          1127  2     srchbaseblk = .idxdat[oib$l_vbn],           ! base vbn of search
;  309          1128  2     srchtopblk = srchbaseblk + .idxdat[oib$l_nblks] - 1, ! top vbn of search
;  310          1129  3     topblkents = (
;  311          1130  3                 IF .idxdat[oib$l_nents] LEQ entsperblk
;  312          1131  3                     THEN .idxdat[oib$l_nents]
;  313          1132  3                     ELSE .idxdat[oib$l_nents] -
;  314          1133  4                         entsperblk*(.idxdat[oib$l_nblks] - 1)
;  315          1134  2                 ),
;  316          1135  2     windowbaseblk = oldctx[ofl$l_winbvbn],      ! Base VBN of window
;  317          1136  2     windowtopblk = oldctx[ofl$l_wintvbn],       ! Top VBN of window
;  318          1137  2     windowblocks = oldctx[ofl$l_winblks],       ! Size in blocks of window
;  319          1138  2     trialent = oldctx[ofl$l_trient],            ! Trial table entry number within block
;  320          1139  2     trialrbn = oldctx[ofl$l_trilrbn],           ! location of entry within window
;  321          1140  2     trialadr = oldctx[ofl$l_triladr] : REF BBLOCK, ! Pointer to entry in table
;  322          1141  2     lolimitent = oldctx[ofl$l_lowent],          ! lowest possible name entry
;  323          1142  2     lolimitrbn = oldctx[ofl$l_lowrbn],          ! Rel. blk number in window
;  324          1143  2     lolimitadr = oldctx[ofl$l_lowadr] : REF BBLOCK, ! and its address in table
;  325          1144  2     hilimitent = oldctx[ofl$l_hient],           ! highest possible name entry
;  326          1145  2     hilimitrbn = oldctx[ofl$l_hirbn],           ! rel. blk number in window
```

```
;  327     1146  2     hilimitadr = oldctx[ofl$l_hiadr] : REF BBLOCK; ! and its address in table
;  328     1147  2
;  329     1148  2 LOCAL
;  330     1149  2     cache_entry,
;  331     1150  2     blockaddr,
;  332     1151  2     trialblockoff,
;  333     1152  2     entryoff,
;  334     1153  2     ch_result,
;  335     1154  2     moveflag,                              ! direction movement control
;  336     1155  2     readwindow;                            ! window read flag
;  337     1156  2
;  338     1157  2 IF .idxdat[oib$l_tbladr] EQL 0             ! If index has not been read
;  339     1158  3 THEN BEGIN
;  340     1159  3     IF .idxdat[oib$l_nblks] EQL 0          ! but if nothing in index
;  341     1160  3         THEN RETURN lbr$_keynotfnd;        ! then return key not found
;  342   P 1161  3     perform(read_n_block(.idxdat[oib$l_vbn],    !  then read it now
;  343     1162  3                          .idxdat[oib$l_nblks]));
;  344     1163  3     idxdat[oib$l_tbladr] = 1;              ! flag index read
;  345     1164  2     END;
;  346     1165  2 moveflag = 0;                              ! reset the direction of last read
;  347     1166  2                                            ! if the current window is
;  348     1167  2 !write('Searching for key !AS', .keydesc);
;  349     1168  2 IF .windowbaseblk GEQU srchbaseblk         ! completely within the search
;  350     1169  2 AND.windowtopblk LEQU srchtopblk           ! range or contains it
;  351     1170  2 THEN
;  352     1171  3     BEGIN                                  ! compare target symbol
;  353     1172  3     perform(find_block(.windowbaseblk+.trialrbn, blockaddr, cache_entry));
;  354     1173  3 !     write('Looking at key !AD',..trialadr[one$b_namlng], trialadr[one$t_name]);
;  355     1174  4     IF (ch_result = CH$COMPARE(.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],
;  356     1175  3                                .trialadr[one$b_namlng], trialadr[one$t_name])) EQL 0
;  357     1176  3     THEN
;  358     1177  4         BEGIN                              ! with last trial
;  359     1178  4         retrfa[rfa$l_vbn]=.trialadr[one$w_modvbn];      ! and if equal
;  360     1179  4         retrfa[rfa$w_offset]=.trialadr[one$w_modbytoff];     ! compute module's
;  361     1180  4         RETURN lbr$_normal;                          ! rfa and return
;  362     1181  4         END
;  363     1182  3     ELSE IF .ch_result LSS 0
;  364     1183  3         THEN
;  365     1184  4             BEGIN                          ! if it is less
;  366     1185  4             IF .trialent EQL 0             ! and if trial
;  367     1186  4             AND.trialrbn EQL 0             ! is first in window
;  368     1187  4             THEN readwindow = -1           ! then set to read backwards
;  369     1188  4             ELSE
;  370     1189  5                 BEGIN                      !  if less
;  371     1190  5                 readwindow = 0;
;  372     1191  5                 sethilimit(trial,,-1)      ! and not first in window
;  373     1192  5                 setlolimit(lolimit,0,0)    ! low limit is the first
;  374     1193  5                 moveflag = -1;             ! set for moved backwards
;  375     1194  4                 END;                       ! and entry before the trial
;  376     1195  4              END                           ! is the highest possible
;  377     1196  3             ELSE
;  378     1197  4                 BEGIN                      ! target is greater
;  379     1198  4                                                    ! than last trial .if
;  380     1199  4                 IF .trialrbn EQL .windowblocks-1   ! last in window, set
;  381     1200  5                 AND(.trialent EQL entsperblk-1
;  382     1201  6                 OR(.windowtopblk EQL srchtopblk
;  383     1202  5                 AND.trialent EQL topblkents-1))
```

```
;  384          1203  4                    THEN readwindow = 1                 ! to read next window
;  385          1204  4                    ELSE
;  386          1205  5                        BEGIN                           ! greater but trial is not
;  387          1206  5                            readwindow = 0;             ! last in window, so
;  388          1207  5                            IF .windowtopblk EQL srchtopblk
;  389          1208  5                            THEN entryoff = topblkents -1
;  390          1209  5                            ELSE entryoff = entsperblk -1;
;  391        P 1210  5                            sethilimit(hilimit,.windowblocks-1,! high limit is top of
;  392          1211  5                                       .entryoff)       ! window and
;  393          1212  5                            setlolimit(trial,,1)        ! lowest possible is trial
;  394          1213  5                            moveflag = 1;               ! set as moved forward
;  395          1214  4                            END;                        ! plus 1
;  396          1215  3                        END;
;  397          1216  3                    END
;  398          1217  2                ELSE
;  399          1218  3                    BEGIN                               ! current window is
;  400          1219  3                        readwindow=1;                   ! no use, set to
;  401          1220  3                        windowtopblk= srchbaseblk-1;    ! read the first
;  402          1221  2                        END;
;  403          1222  2            WHILE true DO                               ! begin an infinite loop
;  404          1223  3            BEGIN
;  405          1224  3                IF .readwindow NEQ 0                    ! another window to be read?
;  406          1225  3                THEN
;  407          1226  4                    BEGIN
;  408          1227  4                    IF .readwindow LSS 0                ! if flag negative, read
;  409          1228  4                                                        ! backwards, provided last was
;  410          1229  4                    THEN IF .moveflag GTR 0             ! not a forward move
;  411          1230  4                        THEN RETURN lbr$_keynotfnd      ! in which
;  412          1231  4                        ELSE
;  413          1232  5                            BEGIN                       ! case name cannot be here
;  414          1233  5                            IF .windowbaseblk EQL srchbaseblk   ! also if back at first window
;  415          1234  5                            THEN RETURN lbr$_keynotfnd;         ! it's all over
;  416          1235  5                            windowbaseblk=.windowbaseblk-       ! otherwise set up the
;  417          1236  5                                          lbr$k_libblocks;      ! window as full
;  418          1237  5                            windowtopblk=.windowtopblk-.windowblocks;   ! since must have been
;  419          1238  5                                                                ! one before
;  420          1239  5                            moveflag=-1;                ! at base of search range
;  421          1240  5                            END                         ! remembering this fact
;  422          1241  5                                                        ! read next window up in
;  423          1242  4                ELSE IF .moveflag LSS 0                 ! the search range, provided
;  424          1243  4                    THEN RETURN lbr$_keynotfnd          ! we haven't reached the
;  425          1244  4                    ELSE
;  426          1245  5                        BEGIN                           ! end already and provided
;  427          1246  5                                                        ! did not reverse last time
;  428          1247  5                        IF .windowtopblk EQL srchtopblk !
;  429          1248  5                        THEN RETURN lbr$_keynotfnd;     ! in which case name cannot be here
;  430          1249  5                        windowbaseblk = .windowtopblk + 1;  ! set block range of
;  431          1250  5                        windowtopblk = MINU(srchtopblk,     ! new window to next and
;  432          1251  6                                    (.windowbaseblk+
;  433          1252  5                                        lbr$k_libblocks-1));    ! perhaps last of range
;  434          1253  5                        moveflag=1;                     ! for next time to suppress backup
;  435          1254  4                        END;
;  436          1255  4                windowblocks = .windowtopblk - .windowbaseblk + 1;  ! calculate number of blocks in window
;  437          1256  4                setlolimit(lolimit,0,0)                 ! low limit on searck is first entry
;  438          1257  4                IF .windowtopblk EQL srchtopblk
;  439          1258  4                THEN entryoff = topblkents -1
;  440          1259  4                ELSE entryoff = entsperblk -1;
```

```
;  441          1260  4              sethilimit(hilimit,.windowblocks-1,.entryoff)                   ! high limit is the last
;  442          1261  4              setrial(hilimit,,)                                          ! initialize the last trial
;  443          1262  4              END;                                                            ! and all set to search it
;  444          1263  3
;  445          1264  3          !
;  446          1265  3          ! we now have a valid window and the highest and lowest possible
;  447          1266  3          ! entries in this window have been set.
;  448          1267  3          ! now compare the target name against these limits
;  449          1268  3          ! and if outside set up for a window read.
;  450          1269  3          ! if between, begin binary search with adjusted limits
;  451          1270  3          ! and a trial half way between.
;  452          1271  3          !
;  453          1272  3                perform(find_block(.windowbaseblk+.hilimitrbn, blockaddr, cache_entry));
;  454          1273  3          !     write('High limit look at !AD',.hilimitadr[one$b_namlng],
;  455          1274  4          !                 hilimitadr[one$t_name]);
;  456          1275  3                IF (ch_result = CH$COMPARE (.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],
;  457          1276  4                    .hilimitadr[one$b_namlng], hilimitadr[one$t_name])) EQL 0
;  458          1277  4                THEN BEGIN                                                                  ! if required name i
;  459          1278  4                    retrfa[rfa$l_vbn] = .hilimitadr[one$w_modvbn];                          ! equal to upper lim
;  460          1279  4                                                                                            ! return the rfa
;  461          1280  4                    retrfa[rfa$w_offset] = .hilimitadr[one$w_modbytoff];                    ! of top limit entry
;  462          1281  4                RETURN lbr$_normal;
;  463          1282  4                END
;  464          1283  4                ELSE BEGIN
;  465          1284  4                     IF .ch_result GTR 0                                                                ! if high limit
;  466          1285  4                                                                                                        ! is greater than top
;  467          1286  5                     THEN readwindow = 1                                                                ! limit, then set to
;  468          1287  5                     ELSE BEGIN                                                                         ! read next up
;  469          1288  5                        IF .hilimitrbn EQL 0                                                            ! if less than high
;  470          1289  5                        AND .hilimitent EQL 0                                                           ! limit, and high limit
;  471          1290  5                                                                                                        ! is first in window
;  472          1291  6                        THEN readwindow = -1                                                            ! then set for first
;  473        P 1292  6                        ELSE BEGIN                                                                      ! window read
;  474          1293  6                             perform(find_block(.windowbaseblk+.lolimitrbn, blockaddr,
;  475          1294  6                                                cache_entry));
;  476          1295  6          !                  write('Lolimit look at !AD',.lolimitadr[one$b_namlng],
;  477          1296  7          !                              lolimitadr[one$t_name]);
;  478          1297  7                              IF (ch_result = CH$COMPARE(.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],       ! ot
;  479          1298  6                                  .lolimitadr[one$b_namlng],                                            ! with low limit and
;  480          1299  7                                  lolimitadr[one$t_name])) EQL 0                                        ! if equal, return
;  481          1300  7                               THEN BEGIN                                                               ! rfa now
;  482          1301  7                                    retrfa[rfa$l_vbn] = .lolimitadr[one$w_modvbn];                      !
;  483          1302  7                                    retrfa[rfa$w_offset] = .lolimitadr[one$w_modbytoff];
;  484          1303  7                                    setrial(lolimit,,)                                                  ! and remember it for next t
;  485          1304  7                                    RETURN lbr$_normal;
;  486          1305  6                                    END
;  487          1306  6                               ELSE IF .ch_result LSS 0                                                 ! not equal but if less, set
;  488          1307  6                                    THEN readwindow = -1                                                ! first window.
;  489          1308  6                                    ELSE IF .lolimitrbn EQL                                             ! if greater than
;  490          1309  7                                                .windowblocks-1                                         ! lolimit, provided
;  491          1310  8                                         AND (.lolimitent EQL entsperblk-1                          ! that is not last
;  492          1311  7                                              OR (.windowtopblk EQL srchtopblk                      ! of window, the
;  493          1312  6                                                  AND .lolimitent EQL topblkents-1))
;  494          1313  6                                         THEN readwindow = 1                                        ! block (set read up)
;  495          1314  5                                         ELSE EXITLOOP;                                             ! we have the window
;  496          1315  4                              END;                                                                  ! containing the re-
;  497          1316  3                     END;                                                                           ! quired name, if
;  498          1317  2          END;                                                                                      ! it is in table
```

```
;  498          1317  2 END;
;  499          1318  2 !++
;  500          1319  2 !       now begin binary search of the window, after adjusting
;  501          1320  2 !       low limit up one and high limit down one; unless they
;  502          1321  2 !       are the same, in which case required name is
;  503          1322  2 !       not here.
;  504          1323  2 !
;  505          1324  2         IF .lolimitadr EQL .hilimitadr
;  506          1325  2         THEN RETURN lbr$_keynotfnd;
;  507          1326  2         setlolimit(lolimit,,1)                                  ! limits are not same, so
;  508          1327  2         IF .lolimitadr NEQ .hilimitadr
;  509          1328  2         THEN sethilimit(hilimit,,-1)                            ! increment low and decr.hig
;  510          1329  2         WHILE .lolimitrbn NEQ .hilimitrbn                       ! loop until we have both
;  511          1330  3         DO BEGIN                                                ! limits on the same block
;  512          1331  3                 trialblockoff = (.hilimitrbn-.lolimitrbn+1)/2;  ! calculate a trial entry
;  513          1332  3                 IF (.lolimitrbn+.trialblockoff) EQL             ! half way between
;  514          1333  4                     (.hilimitrbn-.trialblockoff)                ! that is, middle
;  515          1334  3                 THEN entryoff = (entsperblk+1)/2                ! entry of midblock
;  516          1335  3                                                                 ! or first entry of
;  517          1336  3                 ELSE entryoff = 0;                              ! midpoint higher
;  518          1337  3                                                                 ! block if mid point
;  519          1338  3                 setrial(lolimit,.trialblockoff+.lolimitrbn,.entryoff)   ! is a block boundary
;  520        P 1339  3                 perform(find_block(.windowbaseblk+.trialrbn, blockaddr,
;  521          1340  3                                    cache_entry));
;  522          1341  3 !               write('window search look at !AD',.trialadr[one$b_naming],
;  523          1342  3 !                     trialadr[one$t_name]);
;  524          1343  4                 IF (ch_result = CH$COMPARE(.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],     ! compare re
;  525          1344  3                     .trialadr[one$b_naming], trialadr[one$t_name])) EQL 0   ! and if equal, set
;  526          1345  4                 THEN BEGIN                                      ! up return values
;  527          1346  4                         retrfa[rfa$l_vbn]=.trialadr[one$w_modvbn];
;  528          1347  4                         retrfa[rfa$w_offset]=.trialadr[one$w_modbytoff];
;  529          1348  4                         RETURN lbr$_normal;                             ! and all done
;  530          1349  3                     END;
;  531          1350  3                 IF .ch_result GTR 0                             ! if the required is greater
;  532          1351  4                 THEN BEGIN                                      ! lolimit to trial+1
;  533          1352  4                         setlolimit(trial,,1)
;  534          1353  4                     END
;  535          1354  4                 ELSE BEGIN                                      ! if less, update
;  536          1355  4                     sethilimit(trial,,-1)                       ! high limit to
;  537          1356  3                     END;                                        ! trial-1
;  538          1357  2         END;                                                    ! end of loop
;  539          1358  2 !
;  540          1359  2 !       both limits are now on the same block, start half way
;  541          1360  2 !       between limits and step toward high limit if target is
;  542          1361  2 !       greater, toward low limit if target is less.
;  543          1362  2 !
;  544          1363  2         setrial(lolimit,,(.hilimitent-.lolimitent)/2)           ! set trial at mid
;  545          1364  2         moveflag = 0;                                           ! point entry and clear dire
;  546          1365  3         DO BEGIN                                                ! loop until we hit the limi
;  547        P 1366  3                 perform(find_block(.windowbaseblk+.trialrbn, blockaddr,
;  548          1367  3                                    cache_entry));
;  549          1368  3 !               write('Final look at !AD', .trialadr[one$b_naming],
;  550          1369  3 !                     trialadr[one$t_name]);
;  551          1370  4                 IF (ch_result = CH$COMPARE(.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],     ! if target
;  552          1371  3                     .trialadr[one$b_naming], trialadr[one$t_name])) GTR 0   ! greater than the
;  553          1372  3                 THEN IF .moveflag EQL -1                        ! trial, and were moving
;  554          1373  3                     THEN RETURN lbr$_keynotfnd                          ! backwards - not he
```

```
;  555          1374  4                    ELSE BEGIN                                                  ! otherwise move
;  556          1375  4                       moveflag = 1;                                            ! forward to next
;  557          1376  4                       setrial(trial,,1)                                        ! entry.
;  558          1377  4                         END
;  559          1378  3                 ELSE IF .ch_result EQL 0                                      ! if target is equal to the
;  560          1379  4                 THEN BEGIN                                                    ! then return the parameters
;  561          1380  4                    retrfa[rfa$l_vbn] = .trialadr[one$w_modvbn];
;  562          1381  4                    retrfa[rfa$w_offset] = .trialadr[one$w_modbytoff];
;  563          1382  4                    RETURN lbr$_normal;
;  564          1383  4                      END
;  565          1384  3                 ELSE IF .moveflag EQL 1                                       ! target symbol is
;  566          1385  3                    THEN RETURN lbr$_keynotfnd                                          ! less, so if we wer
;  567          1386  4                    ELSE BEGIN                                                 ! moving forward - name
;  568          1387  4                            moveflag = -1;                                     ! not here. otherwise
;  569          1388  4                            setrial(trial,,-1)                                 ! move back one
;  570          1389  3                         END;
;  571          1390  3              END
;  572          1391  2           UNTIL .trialent EQL .lolimitent                                     ! loop until at the low
;  573          1392  2           OR .trialent EQL .hilimitent;                                       ! or high limit
;  574          1393  2 !
;  575          1394  2 !         trial entry is at one of the limits. check for
;  576          1395  2 !          a match and return values
;  577          1396  2 !
;  578          1397  2 !         write('End of limit check on !AD', .trialadr[one$b_namlng],
;  579          1398  2 !                  trialadr[one$t_name]);
;  580          1399  2           IF CH$EQL(.keydesc[dsc$w_length], .keydesc[dsc$a_pointer],
;  581          1400  2                  .trialadr[one$b_namlng], trialadr[one$t_name])
;  582          1401  3           THEN BEGIN
;  583          1402  3                  retrfa[rfa$l_vbn] = .trialadr[one$w_modvbn];
;  584          1403  3                  retrfa[rfa$w_offset] = .trialadr[one$w_modbytoff];
;  585          1404  3                  RETURN lbr$_normal;
;  586          1405  3                END
;  587          1406  2           ELSE RETURN lbr$_keynotfnd;
;  588          1407  1 END;                                                !Of lbr_old_lkp_key


                                    0FFC 00000          .ENTRY  LBR_OLD_LKP_KEY, Save R2,R3,R4,R5,R6,R7,R8,-; 1089
                                                                R9,R10,R11                                    ;
                          5E      B8  AE  9E 00002          MOVAB   -72(SP), SP                             ;
                          51    0000G CF  D0 00006          MOVL    LBR$GL_CONTROL, R1                      ; 1115
                    50    0A    A1 0000015A 8F  C1 0000B    ADDL3   #346, 10(R1), R0                        ; 1118
                    51    12    A1          01  C3 00014    SUBL3   #1, 18(R1), R1                          ; 1119
                                            05  12 00019    BNEQ    1$                                      ; 1121
                          52              50  D0 0001B    MOVL    R0, R2                                  ; 1122
                                            07  11 0001E    BRB     2$                                      ;
                          56        1C    A0  9E 00020 1$:  MOVAB   28(R0), R6                              ; 1123
                          52              56  D0 00024    MOVL    R6, R2                                  ;
                    24    AE        04    A2  D0 00027 2$:  MOVL    4(R2), 36(SP)                           ; 1125
                    20    AE        10    A2  D0 0002C    MOVL    16(R2), 32(SP)                          ; 1126
                    3C    AE              62  D0 00031    MOVL    (R2), 60(SP)                            ; 1127
                    51    3C    AE  0C    A2  C1 00035    ADDL3   12(R2), 60(SP), R1                      ; 1128
                    2C    AE        FF    A1  9E 0003B    MOVAB   -1(R1), 44(SP)                          ;
                    20    AE        08    A2  D1 00040    CMPL    8(R2), 32(SP)                           ; 1130
                                            07  14 00045    BGTR    3$                                      ;
```

```
                34   AE      08   A2  D0 00047          MOVL    8(R2), 52(SP)                           ; 1131
                                  12  11 0004C          BRB     4$
            51  0C   A2            01  C3 0004E 3$:     SUBL3   #1, 12(R2), R1                          ; 1133
                     51      20   AE  C4 00053          MULL2   32(SP), R1
            56  08   A2            51  C3 00057          SUBL3   R1, 8(R2), R6
                34   AE            56  D0 0005C          MOVL    R6, 52(SP)
                6E      38   A0   9E 00060 4$:          MOVAB   56(R0), (SP)                            ; 1135
                5B      3C   A0   9E 00064              MOVAB   60(R0), R11                             ; 1136
                28   AE     40   A0  9E 00068          MOVAB   64(R0), 40(SP)                          ; 1137
                58      44   A0   9E 0006D              MOVAB   68(R0), R8                              ; 1138
                59      48   A0   9E 00071              MOVAB   72(R0), R9                              ; 1139
                0C   AE     4C   A0  9E 00075          MOVAB   76(R0), 12(SP)                          ; 1140
                14   AE     50   A0  9E 0007A          MOVAB   80(R0), 20(SP)                          ; 1141
                04   AE     54   A0  9E 0007F          MOVAB   84(R0), 4(SP)                           ; 1142
                1C   AE     58   A0  9E 00084          MOVAB   88(R0), 28(SP)                          ; 1143
                5A      5C   A0   9E 00089              MOVAB   92(R0), R10                             ; 1144
                08   AE     60   A0  9E 0008D          MOVAB   96(R0), 8(SP)                           ; 1145
                18   AE     64   A0  9E 00092          MOVAB   100(R0), 24(SP)                         ; 1146
                            14   A2  D5 00097          TSTL    20(R2)                                  ; 1157
                                 19  12 0009A          BNEQ    6$
                            0C   A2  D5 0009C          TSTL    12(R2)                                  ; 1159
                                 03  12 0009F          BNEQ    5$
                               05AA  31 000A1          BRW     86$
                51          0C   A2  D0 000A4 5$:     MOVL    12(R2), R1                              ; 1162
                50              62   D0 000A8          MOVL    (R2), R0
                              0000G  30 000AB          BSBW    READ_N_BLOCK
                27                50   E9 000AE          BLBC    STATUS, 9$
                14   A2             01   D0 000B1          MOVL    #1, 20(R2)                          ; 1163
                            38   AE  D4 000B5 6$:     CLRL    MOVEFLAG                                ; 1165
                3C   AE     00   BE  D1 000B8          CMPL    @0(SP), 60(SP)                          ; 1168
                                 03  1E 000BD          BGEQU   8$
                              0156  31 000BF 7$:        BRW     28$
                2C   AE             6B   D1 000C2 8$:     CMPL    (R11), 44(SP)                       ; 1169
                                    F7   1A 000C6          BGTRU   7$
                52          40   AE  9E 000C8          MOVAB   CACHE_ENTRY, R2                         ; 1172
                51          44   AE  9E 000CC          MOVAB   BLOCKADDR, R1
            50      00   BE            69   C1 000D0          ADDL3   (R9), @0(SP), R0
                              0000G  30 000D5          BSBW    FIND_BLOCK
                5D                50   E9 000D8 9$:     BLBC    STATUS, 14$
                50          04   AC  D0 000DB          MOVL    KEYDESC, R0                             ; 1174
                54          0C   BE  D0 000DF          MOVL    @12(SP), R4                             ; 1175
                51          04   A4  9A 000E3          MOVZBL  4(R4), R1
                55                01   D0 000E7          MOVL    #1, R5
51      00   04   B0                60   2D 000EA          CMPC5   (R0), @4(R0), #0, R1, 5(R4)
                            05   A4       000F0
                                 03  1A 000F2          BGTRU   10$
                55                01   D9 000F4          SBWC    #1, R5
                30   AE             55   D0 000F7 10$:     MOVL    R5, CH_RESULT
                                    03   12 000FB          BNEQ    11$
                                  053A   31 000FD          BRW     84$
                                    70   18 00100 11$:     BGEQ    17$                                 ; 1182
                                    68   D5 00102          TSTL    (R8)                                ; 1185
                                    0A   12 00104          BNEQ    12$
                                    69   D5 00106          TSTL    (R9)                                ; 1186
                                    06   12 00108          BNEQ    12$
                10   AE             01   CE 0010A          MNEGL   #1, READWINDOW                      ; 1187
                                    60   11 0010E          BRB     16$
```

```
                                10   AE   D4 00110 12$:    CLRL    READWINDOW                              ; 1190
                          08    BE   69   D0 00113         MOVL    (R9), @8(SP)                            ; 1191
                    6A          68   01   C3 00117         SUBL3   #1, (R8), (R10)                         ;
                                     0A   18 0011B         BGEQ    13$                                     ;
                    6A    20    AE   01   C3 0011D         SUBL3   #1, 32(SP), (R10)                       ;
              08    BE          69   01   C3 00122         SUBL3   #1, (R9), @8(SP)                        ;
                          52         40   AE 9E 00127 13$: MOVAB   CACHE_ENTRY, R2                        ;
                          51         44   AE 9E 0012B      MOVAB   BLOCKADDR, R1                           ;
                    50    00    BE   08   BE C1 0012F      ADDL3   @8(SP), @0(SP), R0                      ;
                                     0000G 30 00135         BSBW    FIND_BLOCK                              ;
                          22         50   E9 00138 14$:    BLBC    STATUS, 15$                             ;
                    50    24    AE   6A   C5 0013B         MULL3   (R10), 36(SP), R0                       ;
                          18    BE   44 BE40 9E 00140      MOVAB   @BLOCKADDR[R0], @24(SP)                 ;
                                     04   BE D4 00146      CLRL    @4(SP)                                  ; 1192
                                     14   BE D4 00149      CLRL    @20(SP)                                 ;
                          52         40   AE 9E 0014C      MOVAB   CACHE_ENTRY, R2                         ;
                          51         44   AE 9E 00150      MOVAB   BLOCKADDR, R1                           ;
                    50    00    BE   04   BE C1 00154      ADDL3   @4(SP), @0(SP), R0                      ;
                                     0000G 30 0015A         BSBW    FIND_BLOCK                              ;
                          69         50   E9 0015D 15$:    BLBC    STATUS, 23$                             ;
                    50    24    AE   14   BE C5 00160      MULL3   @20(SP), 36(SP), R0                     ;
                          1C    BE   44 BE40 9E 00166      MOVAB   @BLOCKADDR[R0], @28(SP)                 ;
                          38    AE   01   CE 0016C         MNEGL   #1, MOVEFLAG                            ; 1193
                                     28   11 00170 16$:    BRB     19$                                     ; 1182
                    51    28    BE   01   C3 00172 17$:    SUBL3   #1, @40(SP), R1                         ; 1199
                                51   69   D1 00177         CMPL    (R9), R1                                ;
                                     20   12 0017A         BNEQ    20$                                     ;
                    50    20    AE   01   C3 0017C         SUBL3   #1, 32(SP), R0                          ; 1200
                                50   68   D1 00181         CMPL    (R8), R0                                ;
                                     10   13 00184         BEQL    18$                                     ;
                          2C    AE   6B   D1 00186         CMPL    (R11), 44(SP)                           ; 1201
                                     10   12 0018A         BNEQ    20$                                     ;
              56    34    AE   01   C3 0018C         SUBL3   #1, 52(SP), R6                          ; 1202
                                56   68   D1 00191         CMPL    (R8), R6                                ;
                                     06   12 00194         BNEQ    20$                                     ;
                          10    AE   01   D0 00196 18$:    MOVL    #1, READWINDOW                          ; 1203
                                     7A   11 0019A 19$:    BRB     27$                                     ;
                                10   AE   D4 0019C 20$:    CLRL    READWINDOW                              ; 1206
                          2C    AE   6B   D1 0019F         CMPL    (R11), 44(SP)                           ; 1207
                                     07   12 001A3         BNEQ    21$                                     ;
              56    34    AE   01   C3 001A5         SUBL3   #1, 52(SP), ENTRYOFF                    ; 1208
                                     05   11 001AA         BRB     22$                                     ;
              56    20    AE   01   C3 001AC 21$:    SUBL3   #1, 32(SP), ENTRYOFF                    ; 1209
                          08    BE   51   D0 001B1 22$:    MOVL    R1, @8(SP)                              ; 1211
                                6A   56   D0 001B5         MOVL    ENTRYOFF, (R10)                         ;
                                52   40   AE 9E 001B8      MOVAB   CACHE_ENTRY, R2                         ;
                                51   44   AE 9E 001BC      MOVAB   BLOCKADDR, R1                           ;
              50    00    BE   08   BE C1 001C0      ADDL3   @8(SP), @0(SP), R0                      ;
                                     0000G 30 001C6         BSBW    FIND_BLOCK                              ;
                                36   50   E9 001C9 23$:    BLBC    STATUS, 25$                             ;
                    50    24    AE   6A   C5 001CC         MULL3   (R10), 36(SP), R0                       ;
                          18    BE   44 BE40 9E 001D1      MOVAB   @BLOCKADDR[R0], @24(SP)                 ;
                          04    BE   69   D0 001D7         MOVL    (R9), @4(SP)                            ; 1212
                    50          68   01   C1 001DB         ADDL3   #1, (R8), R0                            ;
                          14    BE   50   D0 001DF         MOVL    R0, @20(SP)                             ;
                          20    AE   50   D1 001E3         CMPL    R0, 32(SP)                              ;
                                     08   19 001E7         BLSS    24$                                     ;
```

```
                                  14   BE   D4 001E9          CLRL    @20(SP)                          ;
04   BE              69           01   C1 001EC               ADDL3   #1, (R9), @4(SP)                 ;
                     52           40   AE   9E 001F1   24$:   MOVAB   CACHE_ENTRY, R2                  ;
                     51           44   AE   9E 001F5          MOVAB   BLOCKADDR, R1                    ;
     50    00        BE           04   BE   C1 001F9          ADDL3   @4(SP), @0(SP), R0               ;
                               0000G   30 001FF               BSBW    FIND_BLOCK                       ;
                     01           50   E8 00202        25$:   BLBS    STATUS, 26$                      ;
                                       04 00205               RET                                      ;
     50    24        AE           14   BE   C5 00206   26$:   MULL3   @20(SP), 36(SP), R0              ;
           1C        BE           44 BE40   9E 0020C          MOVAB   @BLOCKADDR[R0], @28(SP)          ;
           38        AE           01   D0 00212               MOVL    #1, MOVEFLAG                     ; 1213
                                  09   11 00216        27$:   BRB     29$                              ; 1168
           10        AE           01   D0 00218        28$:   MOVL    #1, READWINDOW                   ; 1219
     6B    3C        AE           01   C3 0021C               SUBL3   #1, 60(SP), (R11)                ; 1220
                     57           04   AC   D0 00221   29$:   MOVL    KEYDESC, R7                      ; 1274
                                  10   AE   D5 00225   30$:   TSTL    READWINDOW                       ; 1224
                                  03   12 00228               BNEQ    31$                              ;
                                00D3   31 0022A               BRW     39$                              ;
                                  1D   18 0022D        31$:   BGEQ    34$                              ; 1227
                                  38   AE   D5 0022F          TSTL    MOVEFLAG                         ; 1229
                                  03   15 00232               BLEQ    33$                              ;
                                0417   31 00234        32$:   BRW     86$                              ;
           3C        AE           00   BE   D1 00237   33$:   CMPL    @0(SP), 60(SP)                   ; 1233
                                  F6   13 0023C               BEQL    32$                              ;
           00        BE           0A   C2 0023E               SUBL2   #10, @0(SP)                      ; 1235
                     6B           28   BE   C2 00242          SUBL2   @40(SP), (R11)                   ; 1237
           38        AE           01   CE 00246               MNEGL   #1, MOVEFLAG                     ; 1239
                                  28   11 0024A               BRB     36$                              ; 1229
                                  38   AE   D5 0024C   34$:   TSTL    MOVEFLAG                         ; 1242
                                  E3   19 0024F               BLSS    32$                              ;
           2C        AE           6B   D1 00251               CMPL    (R11), 44(SP)                    ; 1247
                                  DD   13 00255               BEQL    32$                              ;
00   BE              6B           01   C1 00257               ADDL3   #1, (R11), @0(SP)                ; 1249
     51    00        BE           09   C1 0025C               ADDL3   #9, @0(SP), R1                   ; 1251
                     50           2C   AE   D0 00261          MOVL    44(SP), R0                       ;
                     51           50   D1 00265               CMPL    R0, R1                           ;
                                  03   1B 00268               BLEQU   35$                              ;
                     50           51   D0 0026A               MOVL    R1, R0                           ;
                     6B           50   D0 0026D        35$:   MOVL    R0, (R11)                        ; 1250
           38        AE           01   D0 00270               MOVL    #1, MOVEFLAG                     ; 1253
     50              6B           00   BE   C3 00274   36$:   SUBL3   @0(SP), (R11), R0                ; 1255
           28        BE           01   A0   9E 00279          MOVAB   1(R0), @40(SP)                   ;
                                  04   BE   D4 0027E          CLRL    @4(SP)                           ; 1256
                                  14   BE   D4 00281          CLRL    @20(SP)                          ;
                     52           40   AE   9E 00284          MOVAB   CACHE_ENTRY, R2                  ;
                     51           44   AE   9E 00288          MOVAB   BLOCKADDR, R1                    ;
     50    00        BE           04   BE   C1 0028C          ADDL3   @4(SP), @0(SP), R0               ;
                               0000G   30 00292               BSBW    FIND_BLOCK                       ;
                     79           50   E9 00295               BLBC    STATUS, 40$                      ;
     50    24        AE           14   BE   C5 00298          MULL3   @20(SP), 36(SP), R0              ;
           1C        BE           44 BE40   9E 0029E          MOVAB   @BLOCKADDR[R0], @28(SP)          ;
           2C        AE           6B   D1 002A4               CMPL    (R11), 44(SP)                    ; 1257
                                  07   12 002A8               BNEQ    37$                              ;
     56    34        AE           01   C3 002AA               SUBL3   #1, 52(SP), ENTRYOFF             ; 1258
                                  05   11 002AF               BRB     38$                              ;
     56    20        AE           01   C3 002B1        37$:   SUBL3   #1, 32(SP), ENTRYOFF             ; 1259
08   BE    28        BE           01   C3 002B6        38$:   SUBL3   #1, @40(SP), @8(SP)              ; 1260
```

```
              6A        56  D0 002BC        MOVL    ENTRYOFF, (R10)                          ;
              52   40   AE  9E 002BF        MOVAB   CACHE_ENTRY, R2                          ;
              51   44   AE  9E 002C3        MOVAB   BLOCKADDR, R1                            ;
     50   00  BE   08   BE  C1 002C7        ADDL3   @8(SP), @0(SP), R0                       ;
                  0000G 30 002CD            BSBW    FIND_BLOCK                               ;
              3E        50  E9 002D0        BLBC    STATUS, 40$                              ;
     50   24  AE        6A  C5 002D3        MULL3   (R10), 36(SP), R0                        ;
          18  BE   44 BE40  9E 002D8        MOVAB   @BLOCKADDR[R0], @24(SP)                  ;
     68       6A        0C  28 002DE        MOVC3   #12, (R10), (R8)                         ; 1261
              52   40   AE  9E 002E2        MOVAB   CACHE_ENTRY, R2                          ;
              51   44   AE  9E 002E6        MOVAB   BLOCKADDR, R1                            ;
     50   00  BE        69  C1 002EA        ADDL3   (R9), @0(SP), R0                         ;
                  0000G 30 002EF            BSBW    FIND_BLOCK                               ;
              60        50  E9 002F2        BLBC    STATUS, 45$                              ;
     50   24  AE        68  C5 002F5        MULL3   (R8), 36(SP), R0                         ;
          0C  BE   44 BE40  9E 002FA        MOVAB   @BLOCKADDR[R0], @12(SP)                  ;
              52   40   AE  9E 00300 39$:   MOVAB   CACHE_ENTRY, R2                          ; 1271
              51   44   AE  9E 00304        MOVAB   BLOCKADDR, R1                            ;
     50   00  BE   08   BE  C1 00308        ADDL3   @8(SP), @0(SP), R0                       ;
                  0000G 30 0030E            BSBW    FIND_BLOCK                               ;
              41        50  E9 00311 40$:   BLBC    STATUS, 45$                              ;
              54   18   BE  D0 00314        MOVL    @24(SP), R4                              ; 1275
              50   04   A4  9A 00318        MOVZBL  4(R4), R0                                ;
              55        01  D0 0031C        MOVL    #1, R5                                   ;
 50  00  04   B7   04   BC  2D 0031F        CMPC5   @KEYDESC, @4(R7), #0, R0, 5(R4)          ;
                   05   A4     00326                                                     ;
                        03  1A 00328        BGTRU   41$                                      ;
              55        01  D9 0032A        SBWC    #1, R5                                   ;
          30  AE        55  D0 0032D 41$:   MOVL    R5, CH_RESULT                            ;
                        03  12 00331        BNEQ    42$                                      ;
                      0304  31 00333        BRW     84$                                      ;
                        03  15 00336 42$:   BLEQ    43$                                      ; 1283
                      009D  31 00338        BRW     51$                                      ;
                   08   BE  D5 0033B 43$:   TSTL    @8(SP)                                   ; 1287
                        04  12 0033E        BNEQ    44$                                      ;
                        6A  D5 00340        TSTL    (R10)                                    ; 1288
                        67  13 00342        BEQL    49$                                      ;
              52   40   AE  9E 00344 44$:   MOVAB   CACHE_ENTRY, R2                          ; 1293
              51   44   AE  9E 00348        MOVAB   BLOCKADDR, R1                            ;
     50   00  BE   04   BE  C1 0034C        ADDL3   @4(SP), @0(SP), R0                       ;
                  0000G 30 00352            BSBW    FIND_BLOCK                               ;
              40        50  E9 00355 45$:   BLBC    STATUS, 47$                              ;
              54   1C   BE  D0 00358        MOVL    @28(SP), R4                              ; 1297
              50   04   A4  9A 0035C        MOVZBL  4(R4), R0                                ;
              55        01  D0 00360        MOVL    #1, R5                                   ; 1298
 50  00  04   B7   04   BC  2D 00363        CMPC5   @KEYDESC, @4(R7), #0, R0, 5(R4)          ;
                   05   A4     0036A                                                     ;
                        03  1A 0036C        BGTRU   46$                                      ;
              55        01  D9 0036E        SBWC    #1, R5                                   ;
          30  AE        55  D0 00371 46$:   MOVL    R5, CH_RESULT                            ;
                        32  12 00375        BNEQ    48$                                      ;
              50   08   AC  D0 00377        MOVL    RETRFA, R0                               ; 1300
              60        64  3C 0037B        MOVZWL  (R4), (R0)                               ;
          04  A0   02   A4  B0 0037E        MOVW    2(R4), 4(R0)                             ; 1301
     68   14  BE        0C  28 00383        MOVC3   #12, @20(SP), (R8)                       ; 1302
              52   40   AE  9E 00388        MOVAB   CACHE_ENTRY, R2                          ;
              51   44   AE  9E 0038C        MOVAB   BLOCKADDR, R1                            ;
```

```
                50     00   BE            69   C1 00390          ADDL3   (R9), @0(SP), R0                                    ;
                                     0000G 30 00395          BSBW    FIND_BLOCK                                           ;
                       74                50   E9 00398 47$:    BLBC    STATUS, 56$                                          ;
                50     24   AE            68   C5 0039B          MULL3   (R8), 36(SP), R0                                     ;
                       0C   BE        44 BE40  9E 003A0          MOVAB   @BLOCKADDR[R0], @12(SP)                              ;
                                        029D  31 003A6          BRW     85$                                                  ; 1303
                                          06  18 003A9 48$:     BGEQ    50$                                                  ; 1305
                       10   AE            01   CE 003AB 49$:    MNEGL   #1, READWINDOW                                       ; 1306
                                          2B  11 003AF          BRB     52$                                                  ;
                50     28   BE            01   C3 003B1 50$:    SUBL3   #1, @40(SP), R0                                      ; 1308
                            50        04   BE  D1 003B6          CMPL    @4(SP), R0                                           ;
                                          23  12 003BA          BNEQ    53$                                                  ;
                50     20   AE            01   C3 003BC          SUBL3   #1, 32(SP), R0                                       ; 1309
                            50        14   BE  D1 003C1          CMPL    @20(SP), R0                                          ;
                                          11  13 003C5          BEQL    51$                                                  ;
                       2C   AE            6B   D1 003C7          CMPL    (R11), 44(SP)                                        ; 1310
                                          12  12 003CB          BNEQ    53$                                                  ;
                50     34   AE            01   C3 003CD          SUBL3   #1, 52(SP), R0                                       ; 1311
                            50        14   BE  D1 003D2          CMPL    @20(SP), R0                                          ;
                                          07  12 003D6          BNEQ    53$                                                  ;
                       10   AE            01   D0 003D8 51$:    MOVL    #1, READWINDOW                                       ; 1312
                                        FE46  31 003DC 52$:     BRW     30$                                                  ;
                       18   BE        1C   BE  D1 003DF 53$:    CMPL    @28(SP), @24(SP)                                     ; 1324
                                          03  12 003E4          BNEQ    54$                                                  ;
                                        0265  31 003E6          BRW     86$                                                  ;
                50     14   BE            01   C1 003E9 54$:    ADDL3   #1, @20(SP), R0                                      ; 1326
                       14   BE            50   D0 003EE          MOVL    R0, @20(SP)                                          ;
                       20   AE            50   D1 003F2          CMPL    R0, 32(SP)                                           ;
                                          06  19 003F6          BLSS    55$                                                  ;
                                    14    BE  D4 003F8          CLRL    @20(SP)                                              ;
                                    04    BE  D6 003FB          INCL    @4(SP)                                               ;
                            52      40    AE  9E 003FE 55$:     MOVAB   CACHE_ENTRY, R2                                      ;
                            51      44    AE  9E 00402          MOVAB   BLOCKADDR, R1                                        ;
                50     00   BE        04   BE  C1 00406          ADDL3   @4(SP), @0(SP), R0                                   ;
                                     0000G 30 0040C          BSBW    FIND_BLOCK                                           ;
                       2F                50   E9 0040F 56$:    BLBC    STATUS, 58$                                          ;
                50     24   AE        14   BE  C5 00412          MULL3   @20(SP), 36(SP), R0                                  ;
                       1C   BE        44 BE40  9E 00418          MOVAB   @BLOCKADDR[R0], @28(SP)                              ;
                       18   BE        1C   BE  D1 0041E          CMPL    @28(SP), @24(SP)                                     ; 1327
                                          0B  13 00423          BEQL    57$                                                  ;
                            08            6A   F4 00425          SOBGEQ  (R10), 57$                                           ; 1328
                6A     20   AE            01   C3 00428          SUBL3   #1, 32(SP), (R10)                                    ;
                                    08    BE  D7 0042D          DECL    @8(SP)                                               ;
                            52      40    AE  9E 00430 57$:     MOVAB   CACHE_ENTRY, R2                                      ;
                            51      44    AE  9E 00434          MOVAB   BLOCKADDR, R1                                        ;
                50     00   BE        08   BE  C1 00438          ADDL3   @8(SP), @0(SP), R0                                   ;
                                     0000G 30 0043E          BSBW    FIND_BLOCK                                           ;
                       74                50   E9 00441 58$:    BLBC    STATUS, 63$                                          ;
                50     24   AE            6A   C5 00444          MULL3   (R10), 36(SP), R0                                    ;
                       18   BE        44 BE40  9E 00449          MOVAB   @BLOCKADDR[R0], @24(SP)                              ;
                       08   BE        04   BE  D1 0044F 59$:    CMPL    @4(SP), @8(SP)                                       ; 1329
                                          03  12 00454          BNEQ    60$                                                  ;
                                        00DA  31 00456          BRW     70$                                                  ;
                50     08   BE        04   BE  C3 00459 60$:    SUBL3   @4(SP), @8(SP), R0                                   ; 1331
                                          50  D6 0045F          INCL    R0                                                   ;
                5B                50        02   C7 00461          DIVL3   #2, R0, TRIALBLOCKOFF                                ;
                51     04   BE            5B   C1 00465          ADDL3   TRIALBLOCKOFF, @4(SP), R1                            ; 1332
```

```
                50      08  BE          5B  C3 0046A          SUBL3   TRIALBLOCKOFF, @8(SP), R0                       ; 1333
                    50      51  D1 0046F          CMPL    R1, R0
                            0B  12 00472          BNEQ    61$
                50      20  AE      01  C1 00474          ADDL3   #1, 32(SP), R0                                  ; 1334
                56          50      02  C7 00479          DIVL3   #2, R0, ENTRYOFF
                            02  11 0047D          BRB     62$
                            56  D4 0047F  61$:    CLRL    ENTRYOFF                                                ; 1336
                69          5B  04  BE  C1 00481  62$:    ADDL3   @4(SP), TRIALBLOCKOFF, (R9)                     ; 1338
                68          56      D0 00486          MOVL    ENTRYOFF, (R8)
                52      40  AE  9E 00489          MOVAB   CACHE_ENTRY, R2
                51      44  AE  9E 0048D          MOVAB   BLOCKADDR, R1
                53      00  BE      69  C1 00491          ADDL3   (R9), @0(SP), R3
                50          53      D0 00496          MOVL    R3, R0
                        0000G   30 00499          BSBW    FIND_BLOCK
                    6B          50      E9 0049C          BLBC    STATUS, 67$
                50      24  AE      68  C5 0049F          MULL3   (R8), 36(SP), R0
                0C      BE  44  BE40    9E 004A4          MOVAB   @BLOCKADDR[R0], @12(SP)
                52      40  AE  9E 004AA          MOVAB   CACHE_ENTRY, R2                                         ; 1340
                51      44  AE  9E 004AE          MOVAB   BLOCKADDR, R1
                50          53      D0 004B2          MOVL    R3, R0
                        0000G   30 004B5          BSBW    FIND_BLOCK
                    4F          50      E9 004B8  63$:    BLBC    STATUS, 67$
                54      0C  BE      D0 004BB          MOVL    @12(SP), R4                                     ; 1344
                50      04  A4      9A 004BF          MOVZBL  4(R4), R0
                55          01      D0 004C3          MOVL    #1, R5
        50      00      04  B7  04  BC  2D 004C6          CMPC5   @KEYDESC, @4(R7), #0, R0, 5(R4)
                            05  A4      004CD
                            03  1A 004CF          BGTRU   64$
                55          01      D9 004D1          SBWC    #1, R5
                30  AE          55      D0 004D4  64$:    MOVL    R5, CH_RESULT
                            03      12 004D8          BNEQ    65$
                        015D    31 004DA          BRW     84$
                            3D      15 004DD  65$:    BLEQ    68$                                             ; 1350
                04  BE          69      D0 004DF          MOVL    (R9), @4(SP)                                    ; 1352
        50      68          01      C1 004E3          ADDL3   #1, (R8), R0
                14  BE          50      D0 004E7          MOVL    R0, @20(SP)
                20  AE          50      D1 004EB          CMPL    R0, 32(SP)
                            08      19 004EF          BLSS    66$
                        14  BE      D4 004F1          CLRL    @20(SP)
04      BE          69          01      C1 004F4          ADDL3   #1, (R9), @4(SP)
                52      40  AE  9E 004F9  66$:    MOVAB   CACHE_ENTRY, R2
                51      44  AE  9E 004FD          MOVAB   BLOCKADDR, R1
        50      00      04  BE      C1 00501          ADDL3   @4(SP), @0(SP), R0
                        0000G   30 00507          BSBW    FIND_BLOCK
                62          50      E9 0050A  67$:    BLBC    STATUS, 73$
        50      24  AE      14  BE  C5 0050D          MULL3   @20(SP), 36(SP), R0
                1C  BE      44  BE40    9E 00513          MOVAB   @BLOCKADDR[R0], @28(SP)
                        FF33    31 00519          BRW     59$                                             ; 1350
                08  BE          69      D0 0051C  68$:    MOVL    (R9), @8(SP)                                    ; 1355
        6A          68          01      C3 00520          SUBL3   #1, (R8), (R10)
                            0A      18 00524          BGEQ    69$
        6A      20  AE          01      C3 00526          SUBL3   #1, 32(SP), (R10)
08      BE          69          01      C3 0052B          SUBL3   #1, (R9), @8(SP)
                        FEFD    31 00530  69$:    BRW     57$
                69          04  BE      D0 00533  70$:    MOVL    @4(SP), (R9)                                    ; 1363
        50          6A      14  BE      C3 00537          SUBL3   @20(SP), (R10), R0
                50          02      C6 0053C          DIVL2   #2, R0
```

```
                68          50      14  BE  C1 0053F          ADDL3   @20(SP), R0, (R8)                                    ;
                                        0C  18 00544          BGEQ    71$                                                  ;
                68      20  AE          01  C3 00546          SUBL3   #1, 32(SP), (R8)                                     ;
                69      04  BE          01  C3 0054B          SUBL3   #1, @4(SP), (R9)                                     ;
                                        0D  11 00550          BRB     72$                                                  ;
                        20  AE          68  D1 00552 71$:     CMPL    (R8), 32(SP)                                         ;
                                        07  19 00556          BLSS    72$                                                  ;
                                        68  D4 00558          CLRL    (R8)                                                 ;
                69      04  BE          01  C1 0055A          ADDL3   #1, @4(SP), (R9)                                     ;
                            52      40  AE  9E 0055F 72$:     MOVAB   CACHE_ENTRY, R2                                      ;
                            51      44  AE  9E 00563          MOVAB   BLOCKADDR, R1                                        ;
                50      00  BE          69  C1 00567          ADDL3   (R9), @0(SP), R0                                     ;
                                    0000G 30 0056C            BSBW    FIND_BLOCK                                           ;
                            1E          50  E9 0056F 73$:     BLBC    STATUS, 75$                                          ;
                50      24  AE          68  C5 00572          MULL3   (R8), 36(SP), R0                                     ;
                        0C  BE      44 BE40  9E 00577          MOVAB   @BLOCKADDR[R0], @12(SP)                              ;
                                    38  AE  D4 0057D          CLRL    MOVEFLAG                                             ; 1364
                            52      40  AE  9E 00580 74$:     MOVAB   CACHE_ENTRY, R2                                      ; 1367
                            51      44  AE  9E 00584          MOVAB   BLOCKADDR, R1                                        ;
                50      00  BE          69  C1 00588          ADDL3   (R9), @0(SP), R0                                     ;
                                    0000G 30 0058D            BSBW    FIND_BLOCK                                           ;
                            78          50  E9 00590 75$:     BLBC    STATUS, 81$                                          ;
                            54      0C  BE  D0 00593          MOVL    @12(SP), R4                                          ; 1371
                            50      04  A4  9A 00597          MOVZBL  4(R4), R0                                            ;
                            55          01  D0 0059B          MOVL    #1, R5                                               ;
        50      00      04  B7      04  BC  2D 0059E          CMPC5   @KEYDESC, @4(R7), #0, R0, 5(R4)                      ;
                                    05  A4     005A5                                                                       ;
                                        03  1A 005A7          BGTRU   76$                                                  ;
                            55          01  D9 005A9          SBWC    #1, R5                                               ;
                        30  AE          55  D0 005AC 76$:     MOVL    R5, CH_RESULT                                        ;
                                        33  15 005B0          BLEQ    78$                                                  ;
            FFFFFFFF    8F      38  AE  D1 005B2          CMPL    MOVEFLAG, #-1                                        ; 1372
                                        2F  13 005BA          BEQL    79$                                                  ;
                        38  AE          01  D0 005BC          MOVL    #1, MOVEFLAG                                         ; 1375
        50                  68          01  C1 005C0          ADDL3   #1, (R8), R0                                         ; 1376
                            68          50  D0 005C4          MOVL    R0, (R8)                                             ;
                        20  AE          50  D1 005C7          CMPL    R0, 32(SP)                                           ;
                                        04  19 005CB          BLSS    77$                                                  ;
                                        68  D4 005CD          CLRL    (R8)                                                 ;
                                        69  D6 005CF          INCL    (R9)                                                 ;
                            52      40  AE  9E 005D1 77$:     MOVAB   CACHE_ENTRY, R2                                      ;
                            51      44  AE  9E 005D5          MOVAB   BLOCKADDR, R1                                        ;
        50      00      BE              69  C1 005D9          ADDL3   (R9), @0(SP), R0                                     ;
                                    0000G 30 005DE            BSBW    FIND_BLOCK                                           ;
                            2A          50  E8 005E1          BLBS    STATUS, 82$                                          ;
                                        04  005E4              RET                                                          ;
                                        53  13 005E5 78$:     BEQL    84$                                                  ; 1378
                            01      38  AE  D1 005E7          CMPL    MOVEFLAG, #1                                         ; 1384
                                        61  13 005EB 79$:     BEQL    86$                                                  ;
                        38  AE          01  CE 005ED          MNEGL   #1, MOVEFLAG                                         ; 1387
                            07          68  F4 005F1          SOBGEQ  (R8), 80$                                            ; 1388
        68      20  AE              01  C3 005F4          SUBL3   #1, 32(SP), (R8)                                     ;
                                        69  D7 005F9          DECL    (R9)                                                 ;
                            52      40  AE  9E 005FB 80$:     MOVAB   CACHE_ENTRY, R2                                      ;
                            51      44  AE  9E 005FF          MOVAB   BLOCKADDR, R1                                        ;
        50      00      BE              69  C1 00603          ADDL3   (R9), @0(SP), R0                                     ;
                                    0000G 30 00608            BSBW    FIND_BLOCK                                           ;
```

```
                        47              50  E9 0060B 81$:    BLBC    STATUS, 87$                                     ;
                50      24      AE      68  C5 0060E 82$:    MULL3   (R8), 36(SP), R0                               ;
                        0C      BE  44 BE40  9E 00613         MOVAB   @BLOCKADDR[R0], @12(SP)                         ;
                        14      BE      68  D1 00619         CMPL    (R8), @20(SP)                                   ; 1391
                                        08  13 0061D         BEQL    83$                                             ;
                                6A      68  D1 0061F         CMPL    (R8), (R10)                                     ; 1392
                                        03  13 00622         BEQL    83$                                             ;
                                    FF59    31 00624         BRW     74$                                             ;
                                54      0C  BE  D0 00627 83$:    MOVL    @12(SP), R4                                 ; 1400
                                50      04  A4  9A 0062B         MOVZBL  4(R4), R0                                   ;
        50              00      04  B7  04  BC  2D 0062F         CMPC5   @KEYDESC, @4(R7), #0, R0, 5(R4)             ;
                                    05  A4      00636                                                               ;
                                        14  12 00638         BNEQ    86$                                             ;
                                50      08  AC  D0 0063A 84$:    MOVL    RETRFA, R0                                  ; 1402
                                60          64  3C 0063E         MOVZWL  (R4), (R0)                                  ;
                        04      A0      02  A4  B0 00641         MOVW    2(R4), 4(R0)                                ; 1403
                                50 00000000G    8F  D0 00646 85$:    MOVL    #LBR$_NORMAL, R0                        ; 1406
                                                    04 0064D         RET                                             ;
                                50 00000000G    8F  D0 0064E 86$:    MOVL    #LBR$_KEYNOTFND, R0                     ;
                                                    04 00655 87$:    RET                                             ; 1407

; Routine Size:  1622 bytes,    Routine Base:  $CODE$ + 00C1
```

```
;  590          1408  1 GLOBAL ROUTINE lbr_old_get_idx (index, user_routine, match_desc) =
;  591          1409  2 BEGIN
;  592          1410  2 !
;  593          1411  2 ! This routine calls the specified user routine for each entry
;  594          1412  2 ! in the index
;  595          1413  2 !
;  596          1414  2
;  597        P 1415  2 perform(travers_old_idx(.index, (IF .match_desc NEQ 0
;  598        P 1416  2                                  THEN check_wild
;  599        P 1417  2                                  ELSE call_user),
;  600          1418  2                              .user_routine, .match_desc));
;  601          1419  2 RETURN true
;  602          1420  1 END;                                        !Of lbr_old_get_idx


                                        0000 00000          .ENTRY  LBR_OLD_GET_IDX, Save nothing                   ; 1408
                              7E    08  AC  7D 00002          MOVQ    USER_ROUTINE, -(SP)                             ; 1418
                                    0C  AC  D5 00006          TSTL    MATCH_DESC                                      ;
                                        07  13 00009          BEQL    1$                                              ;
                              50  0000V CF  9E 0000B          MOVAB   CHECK_WILD, R0                                  ;
                                        05  11 00010          BRB     2$                                              ;
                              50  0000V CF  9E 00012 1$:      MOVAB   CALL_USER, R0                                   ;
                                        50  DD 00017 2$:      PUSHL   R0                                              ;
                                    04  AC  DD 00019          PUSHL   INDEX                                           ;
                        0000V CF                04  FB 0001C          CALLS   #4, TRAVERS_OLD_IDX                     ;
                              03                50  E9 00021          BLBC    STATUS, 3$                              ;
                              50                01  D0 00024          MOVL    #1, R0                                  ; 1419
                                                    04 00027 3$:      RET                                             ; 1420

; Routine Size:  40 bytes,    Routine Base:  $CODE$ + 0717
```

```
;  604          1421  1 GLOBAL ROUTINE lbr_old_src_idx (index, rfa, user_routine) =
;  605          1422  2 BEGIN
;  606          1423  2 !
;  607          1424  2 ! This routine searches the index for the given RFA and calls the
;  608          1425  2 ! user routine for each entry that matches.
;  609          1426  2 !
;  610          1427  2
;  611          1428  2 perform(travers_old_idx(.index, check_rfa, .user_routine, .rfa));
;  612          1429  2 RETURN true
;  613          1430  1 END;                                          !Of lbr_old_src_idx


                                    0000 00000         .ENTRY  LBR_OLD_SRC_IDX, Save nothing                   ; 1421
                                08  AC  DD 00002         PUSHL   RFA                                             ; 1428
                                0C  AC  DD 00005         PUSHL   USER_ROUTINE                                    ;
                              0000V CF  9F 00008         PUSHAB  CHECK_RFA                                       ;
                                04  AC  DD 0000C         PUSHL   INDEX                                           ;
                         0000V  CF        04 FB 0000F    CALLS   #4, TRAVERS_OLD_IDX                             ;
                                03        50 E9 00014    BLBC    STATUS, 1$                                      ;
                                50        01 D0 00017    MOVL    #1, R0                                          ; 1429
                                          04 0001A 1$:   RET                                                     ; 1430

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 073F
```

```
;  615          1431  1 ROUTINE check_wild (entry, user_routine, match_desc) =
;  616          1432  2 BEGIN
;  617          1433  2 !---
;  618          1434  2 !       Called by traverse for each entry in the index.  Check to
;  619          1435  2 !       see if current entry matches the match_desc.  Call user if so.
;  620          1436  2 !
;  621          1437  2 ! Inputs:
;  622          1438  2 !
;  623          1439  2 !       entry = Address of key entry
;  624          1440  2 !       user_routine = Address of user action routine
;  625          1441  2 !       match_desc = string descriptor for match string
;  626          1442  2 !
;  627          1443  2 !---
;  628          1444  2 MAP
;  629          1445  2     entry : REF BBLOCK,
;  630          1446  2     match_desc : REF BBLOCK;
;  631          1447  2
;  632          1448  3 IF (fmg$match_name (.entry [one$b_naming], entry [one$t_name],
;  633          1449  3                         .match_desc [dsc$w_length],
;  634          1450  3                         .match_desc [dsc$a_pointer])
;  635          1451  3         OR CH$EQL (.match_desc [dsc$w_length], entry [one$t_name],
;  636          1452  3                         .match_desc [dsc$w_length],
;  637          1453  3                         .match_desc [dsc$a_pointer]))
;  638          1454  2     THEN perform (call_user (.entry, .user_routine));
;  639          1455  2 RETURN true
;  640          1456  1 END;                                           !Of check_wild


                                        03FC 00000 CHECK_WILD:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 1431
                            57      0C  AC  D0 00002       MOVL    MATCH_DESC, R7                          ; 1450
                            56      04  AC  D0 00006       MOVL    ENTRY, R6                               ; 1448
                            53      05  A6  9E 0000A       MOVAB   5(R6), R3                               ;
                            55      04  A7  D0 0000E       MOVL    4(R7), R5                               ;
                            54          67  3C 00012       MOVZWL  (R7), R4                                ;
                            52      04  A6  9A 00015       MOVZBL  4(R6), R2                               ;
                                  0000G 30  00019         BSBW    FMG$MATCH_NAME                          ;
                            08          50  E8 0001C       BLBS    R0, 1$                                  ;
                  04  B7        05  A6      67  29 0001F   CMPC3   (R7), 5(R6), @4(R7)                    ; 1451
                                          0D  12 00025     BNEQ    2$                                      ;
                                      08  AC  DD 00027 1$:  PUSHL   USER_ROUTINE                           ; 1454
                                          56  DD 0002A     PUSHL   R6                                      ;
                           0000V  CF        02  FB 0002C   CALLS   #2, CALL_USER                           ;
                            03            50  E9 00031     BLBC    STATUS, 3$                              ;
                            50            01  D0 00034 2$:  MOVL    #1, R0                                  ; 1455
                                              04 00037 3$:  RET                                             ; 1456

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 075A
```

```
;  642        1457  1 ROUTINE check_rfa (entry, user_routine, rfa) =
;  643        1458  2 BEGIN
;  644        1459  2 !
;  645        1460  2 ! This routine checks if the RFA of the entry matches the given RFA
;  646        1461  2 ! and calls the user back if so.
;  647        1462  2 !
;  648        1463  2
;  649        1464  2 MAP
;  650        1465  2     entry : REF BBLOCK,
;  651        1466  2     rfa : REF BBLOCK;
;  652        1467  2
;  653        1468  2 IF .entry[one$w_modvbn] EQL .rfa[rfa$l_vbn]
;  654        1469  2 AND .entry[one$w_modbytoff] EQL .rfa[rfa$w_offset]
;  655        1470  2 THEN perform (call_user (.entry, .user_routine));
;  656        1471  2 RETURN true
;  657        1472  1 END;                                              !Of check_rfa



                                     0000 00000 CHECK_RFA:
                                                      .WORD   Save nothing                         ; 1457
                      50        04  AC  D0 00002      MOVL    ENTRY, R0                            ; 1468
                      51        0C  AC  D0 00006      MOVL    RFA, R1                              ;
     61       60      10            00  ED 0000A      CMPZV   #0, #16, (R0), (R1)                  ;
                                    14  12 0000F      BNEQ    1$                                   ;
               04     A1        02  A0  B1 00011      CMPW    2(R0), 4(R1)                         ; 1469
                                    0D  12 00016      BNEQ    1$                                   ;
                                08  AC  DD 00018      PUSHL   USER_ROUTINE                         ; 1470
                                    50  DD 0001B      PUSHL   R0                                   ;
             0000V    CF            02  FB 0001D      CALLS   #2, CALL_USER                        ;
                      03            50  E9 00022      BLBC    STATUS, 2$                           ;
                      50            01  D0 00025 1$:  MOVL    #1, R0                               ; 1471
                                        04 00028 2$:  RET                                          ; 1472

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 0792
```

```
;  659          1473  1 ROUTINE call_user (entry, user_routine) =
;  660          1474  2 BEGIN
;  661          1475  2 !
;  662          1476  2 ! This routine calls the user routine for a given entry
;  663          1477  2 !
;  664          1478  2 MAP
;  665          1479  2     entry : REF BBLOCK;
;  666          1480  2
;  667          1481  2 LOCAL
;  668          1482  2     desc : BBLOCK[dsc$c_s_bln],
;  669          1483  2     localrfa : BBLOCK[rfa$c_length];
;  670          1484  2
;  671          1485  2 BIND
;  672          1486  2     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK;
;  673          1487  2
;  674          1488  2 localrfa[rfa$l_vbn] = .entry[one$w_modvbn];
;  675          1489  2 localrfa[rfa$w_offset] = .entry[one$w_modbytoff];
;  676          1490  2 desc[dsc$w_length] = .entry[one$b_nam[ng];
;  677          1491  2 desc[dsc$a_pointer] = entry[one$t_name];
;  678          1492  2 perform ((.user_routine) (desc, localrfa));
;  679          1493  2 context [ctx$v_found1] = true;
;  680          1494  2 RETURN true
;  681          1495  1 END;                                    !Of call_user


                                0004 00000 CALL_USER:
                                           .WORD   Save R2                                 ; 1473
                     5E         0C   C2 00002          SUBL2   #12, SP                    ;
                     50   0000G   CF   D0 00005          MOVL    LBR$GL_CONTROL, R0         ; 1486
                     52     0E    A0   D0 0000A          MOVL    14(R0), R2                 ;
                     50     04    AC   D0 0000E          MOVL    ENTRY, R0                  ; 1488
                     7E           60   3C 00012          MOVZWL  (R0), LOCALRFA             ;
               04    AE     02    A0   B0 00015          MOVW    2(R0), LOCALRFA+4          ; 1489
               08    AE     04    A0   9B 0001A          MOVZBW  4(R0), DESC                ; 1490
               0C    AE     05    A0   9E 0001F          MOVAB   5(R0), DESC+4              ; 1491
                                5E   DD 00024          PUSHL   SP                         ; 1492
                                0C   AE 9F 00026          PUSHAB  DESC                    ;
               08    BC           02   FB 00029          CALLS   #2, @USER_ROUTINE          ;
                     08           50   E9 0002D          BLBC    STATUS, 1$                 ;
               04    A2     40    8F   88 00030          BISB2   #64, 4(R2)                 ; 1493
                     50           01   D0 00035          MOVL    #1, R0                     ; 1494
                                     04 00038 1$:       RET                                ; 1495

; Routine Size:  57 bytes,    Routine Base:  $CODE$ + 07BB
```

```
;  683          1496  1 ROUTINE travers_old_idx (index, action_routine, user_routine, rfa) =
;  684          1497  2 BEGIN
;  685          1498  2 !
;  686          1499  2 ! This routine calls the given action routine for each entry in the index.
;  687          1500  2 !
;  688          1501  2 MAP
;  689          1502  2     rfa : REF BBLOCK;
;  690          1503  2
;  691          1504  2 BIND
;  692          1505  2     context = .lbr$gl_control[lbr$l_ctxptr] : BBLOCK,    !Context block
;  693          1506  2     header = .lbr$gl_control[lbr$l_hdrptr] : BBLOCK,     !Library header
;  694          1507  2     oldctx = header[ohd$t_oldctx] : BBLOCK,              !Old library context block
;  695          1508  3     idxdat = (
;  696          1509  3                 IF .index EQL 1
;  697          1510  3                 THEN oldctx[ofl$l_mntvbn]
;  698          1511  3                 ELSE oldctx[ofl$l_gstvbn]
;  699          1512  2                 ) : BBLOCK,
;  700          1513  2     entrysize = .idxdat[oib$l_esiz],                     !Size of an entry
;  701          1514  2     entsperblk = .idxdat[oib$l_entpblk],                 !Number of entries in a block
;  702          1515  3     topblkents = (
;  703          1516  3                 IF .idxdat[oib$l_nents] LEQ entsperblk
;  704          1517  3                 THEN .idxdat[oib$l_nents]
;  705          1518  4                 ELSE (.idxdat[oib$l_nents]
;  706          1519  4                  - entsperblk*(.idxdat[oib$l_nblks] - 1))
;  707          1520  2                 );
;  708          1521  2
;  709          1522  2 LOCAL
;  710          1523  2     cache_entry,
;  711          1524  2     blkadr : REF VECTOR[,BYTE];
;  712          1525  2
;  713          1526  2 !
;  714          1527  2 ! Read in index if necessary
;  715          1528  2 !
;  716          1529  2 IF .idxdat[oib$l_tbladr] EQL 0                    ! If index has not been read
;  717          1530  3 THEN BEGIN
;  718          1531  3     IF .idxdat[oib$l_nblks] EQL 0                  ! If index is empty
;  719          1532  3         THEN return true;                          !  then all done
;  720        P 1533  3     perform(read_n_block(.idxdat[oib$l_vbn],       !  then read it now
;  721          1534  3                 .idxdat[oib$l_nblks]));
;  722          1535  3     idxdat[oib$l_tbladr] = 1;                      ! flag index read
;  723          1536  2     END;
;  724          1537  2
;  725          1538  2 IF .idxdat[oib$l_nents] GTRU entsperblk                  !If at least one full block
;  726          1539  2 THEN
;  727          1540  2     INCRU i FROM 0 TO .idxdat[oib$l_nblks] - 2
;  728          1541  3     DO BEGIN
;  729          1542  3         perform(find_block(.idxdat[oib$l_vbn]+.i, blkadr, cache_entry)); !Find block in memory
;  730          1543  3 !
;  731          1544  3 ! Call action routine for all entries in block
;  732          1545  3 !
;  733          1546  3         INCRU j FROM 0 TO entsperblk - 1
;  734          1547  3             DO IF NOT (.action_routine) (blkadr[.j*entrysize],
;  735          1548  3                                          .user_routine, .rfa)
;  736          1549  3                 THEN RETURN true;
;  737          1550  2         END;
;  738          1551  2 !
;  739          1552  2 ! Now do the partial block (if it exists)
```

```
;  740          1553  2 !
;  741          1554  2 IF topblkents GTR 0
;  742          1555  3 THEN BEGIN
;  743        P 1556  3     perform(find_block(.idxdat[oib$l_vbn] + .idxdat[oib$l_nblks] - 1,
;  744          1557  3                        blkadr, cache_entry));
;  745          1558  3     INCRU i FROM 0 TO topblkents - 1
;  746          1559  3         DO IF NOT (.action_routine) (blkadr[.i*entrysize],
;  747          1560  3                                      .user_routine, .rfa)
;  748          1561  3                    THEN RETURN true;
;  749          1562  2     END;
;  750          1563  2 RETURN true
;  751          1564  1 END;                                              !Of travers_old_idx


                                    0FFC 00000 TRAVERS_OLD_IDX:
                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1496
                                    5E      08 C2 00002         SUBL2   #8, SP
                                    50  0000G   CF D0 00005         MOVL    LBR$GL_CONTROL, R0      ; 1505
                              50    0A    A0 0000015A 8F C1 0000A   ADDL3   #346, T0(R0), R0        ; 1507
                                    01    04 AC D1 00013         CMPL    INDEX, #1               ; 1509
                                          03 13 00017         BEQL    1$
                                    50      1C C0 00019         ADDL2   #28, R0                 ; 1511
                                    53      50 D0 0001C 1$:     MOVL    R0, R3
                                    59    04 A3 D0 0001F        MOVL    4(R3), R9               ; 1513
                                    56    10 A3 D0 00023        MOVL    16(R3), R6              ; 1514
                                    56    08 A3 D1 00027        CMPL    8(R3), R6               ; 1516
                                          06 14 0002B         BGTR    2$
                                    58    08 A3 D0 0002D        MOVL    8(R3), R8               ; 1517
                                          10 11 00031         BRB     3$
                              50    0C    A3    01 C3 00033 2$: SUBL3   #1, 12(R3), R0          ; 1519
                                    50      56 C4 00038         MULL2   R6, R0
                              50    08    A3    50 C3 0003B     SUBL3   R0, 8(R3), R0
                                    58      50 D0 00040         MOVL    R0, R8                  ; 1518
                                      14  A3 D5 00043 3$:     TSTL    20(R3)                  ; 1529
                                          16 12 00046         BNEQ    4$
                                      0C  A3 D5 00048         TSTL    12(R3)                  ; 1531
                                          5C 13 0004B         BEQL    10$
                                    51    0C  A3 D0 0004D       MOVL    12(R3), R1              ; 1534
                                    50      63 D0 00051         MOVL    (R3), R0
                                        0000G 30 00054         BSBW    READ_N_BLOCK
                                    64      50 E9 00057         BLBC    STATUS, 11$
                                  14    A3    01 D0 0005A       MOVL    #1, 20(R3)              ; 1535
                                    56    08  A3 D1 0005E 4$:   CMPL    8(R3), R6               ; 1538
                                          43 1B 00062         BLEQU   9$
                              57    0C    A3    02 C3 00064     SUBL3   #2, 12(R3), R7          ; 1540
                                          54 D4 00069         CLRL    I                       ; 1548
                                          35 11 0006B         BRB     8$
                                    52      6E 9E 0006D 5$:     MOVAB   CACHE_ENTRY, R2         ; 1542
                                    51    04  AE 9E 00070       MOVAB   BLKADR, R1
                              50          54  63 C1 00074       ADDL3   (R3), I, R0
                                        0000G 30 00078         BSBW    FIND_BLOCK
                                    68      50 E9 0007B         BLBC    STATUS, 15$
                                    55    FF  A6 9E 0007E       MOVAB   -1(R6), R5              ; 1546
                                          52 D4 00082         CLRL    J                       ; 1547
```

```
                          15  11 00084          BRB     7$                                  ;
               7E     0C  AC  7D 00086 6$:      MOVQ    USER_ROUTINE, -(SP)                 ; 1548
      50       52         59  C5 0008A          MULL3   R9, J, R0                           ; 1547
                  0C BE40     9F 0008E          PUSHAB  @BLKADR[R0]                         ;
           08  BC         03  FB 00092          CALLS   #3, @ACTION_ROUTINE                 ;
               4A         50  E9 00096          BLBC    R0, 14$                             ;
                          52  D6 00099          INCL    J                                   ;
               55         52  D1 0009B 7$:      CMPL    J, R5                               ;
                          E6  1B 0009E          BLEQU   6$                                  ;
                          54  D6 000A0          INCL    I                                   ; 1540
               57         54  D1 000A2 8$:      CMPL    I, R7                               ;
                          C6  1B 000A5          BLEQU   5$                                  ;
                          58  D5 000A7 9$:      TSTL    R8                                  ; 1554
                          38  15 000A9 10$:     BLEQ    14$                                 ;
               52         6E  9E 000AB          MOVAB   CACHE_ENTRY, R2                     ; 1557
               51     04  AE  9E 000AE          MOVAB   BLKADR, R1                          ;
      53       63     0C  A3  C1 000B2          ADDL3   12(R3), (R3), R3                    ;
               50     FF  A3  9E 000B7          MOVAB   -1(R3), R0                          ;
                    0000G 30 000BB              BSBW    FIND_BLOCK                          ;
               25         50  E9 000BE 11$:     BLBC    STATUS, 15$                         ;
               53     FF  A8  9E 000C1          MOVAB   -1(R8), R3                          ; 1558
                          52  D4 000C5          CLRL    I                                   ; 1559
                          15  11 000C7          BRB     13$                                 ;
               7E     0C  AC  7D 000C9 12$:     MOVQ    USER_ROUTINE, -(SP)                 ; 1560
      50       52         59  C5 000CD          MULL3   R9, I, R0                           ; 1559
                  0C BE40     9F 000D1          PUSHAB  @BLKADR[R0]                         ;
           08  BC         03  FB 000D5          CALLS   #3, @ACTION_ROUTINE                 ;
               07         50  E9 000D9          BLBC    R0, 14$                             ;
                          52  D6 000DC          INCL    I                                   ;
               53         52  D1 000DE 13$:     CMPL    I, R3                               ;
                          E6  1B 000E1          BLEQU   12$                                 ;
               50         01  D0 000E3 14$:     MOVL    #1, R0                              ; 1563
                              04 000E6 15$:     RET                                         ; 1564

; Routine Size:  231 bytes,    Routine Base:  $CODE$ + 07F4


;  752        1565  1 END                                              !Of module
;  753        1566  0 ELUDOM



;                              PSECT SUMMARY
;
;       Name                 Bytes                          Attributes
;
;  $CODE$                     2267  NOVEC,NOWRT, RD , EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  .  ABS  .                     0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)



;                        Library Statistics
;
```

```
;                                  --------  Symbols  --------      Pages       Processing
;       File                       Total     Loaded   Percent     Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1    9776       13         0        581       00:01.0




;                                     COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:OLDLIB/OBJ=OBJ$:OLDLIB MSRC$:OLDLIB/UPDATE=(ENH$:OLDLIB)

; Size:         2267 code + 0 data bytes
; Run Time:       00:56.6
; Elapsed Time:   01:57.3
; Lines/CPU Min:   1658
; Lexemes/CPU-Min: 25860
; Memory Used:  571 pages
; Compilation Complete
```

OLDLIB
LIS

OPENCLOSE
LIS

OUTPUTHLP
LIS

LBRMSG
LIS